Programming

Programming

Remote Control

- Overview
- Setting up Analyzer
- Performing Calibration
- Making Measurement
- Reading-Writing Measurement Data
- Analyzing Data
- Saving and Recalling
- Communication with External Devices
- Status Reporting System
- Sample Programs

VBA Programming (Embedded VBA)

- Introduction to VBA Programming
- Operation Basics
- Controlling E5061B
- Controlling Peripherals
- Application Programs
- Complex Operation Library
- Waveform Analysis Library

Command Reference

- Notational Conventions
- COM Object Model
- Command Finder

Remote Control

Remote Control

- Overview
- Setting up Analyzer
- Performing Calibration
- Making Measurement
- Reading-Writing Measurement Data
- Analyzing Data
- Saving and Recalling
- Communication with External Devices
- Status Reporting System
- Sample Programs

Overview

Overview

- Types of remote control system
- GPIB remote control system
- LAN remote control system
- USB Remote Control System
- Sending SCPI command messages

Types of remote control system

Depending on the system controller and the interface, you can configure 5 types of remote control system as shown in the table below.

System controller	Interface	Overview
External controller (external computer such as PC and workstation)	GPIB (talker/listener mode)	System to control the E5061B and other devices connected via GPIB from the external controller. For more information, refer to GPIB remote control system.
	LAN	System to control the E5061B and other devices connected via LAN from the external controller. For more information, refer to LAN remote control system.
	USB	System to control the E5061B and other devices connected via USB from the external controller. For more information, refer to USB Remote Control System.
E5061B	None (Internal Connection)	System to control the E5061B itself using built-in E5061B VBA.
	GPIB (system controller mode)	System to control the E5061B itself and external devices connected via GPIB using built-in E5061B VBA.

Other topics about Overview

GPIB remote control system

- About GPIB
- System Configuration

Other topics about Overview

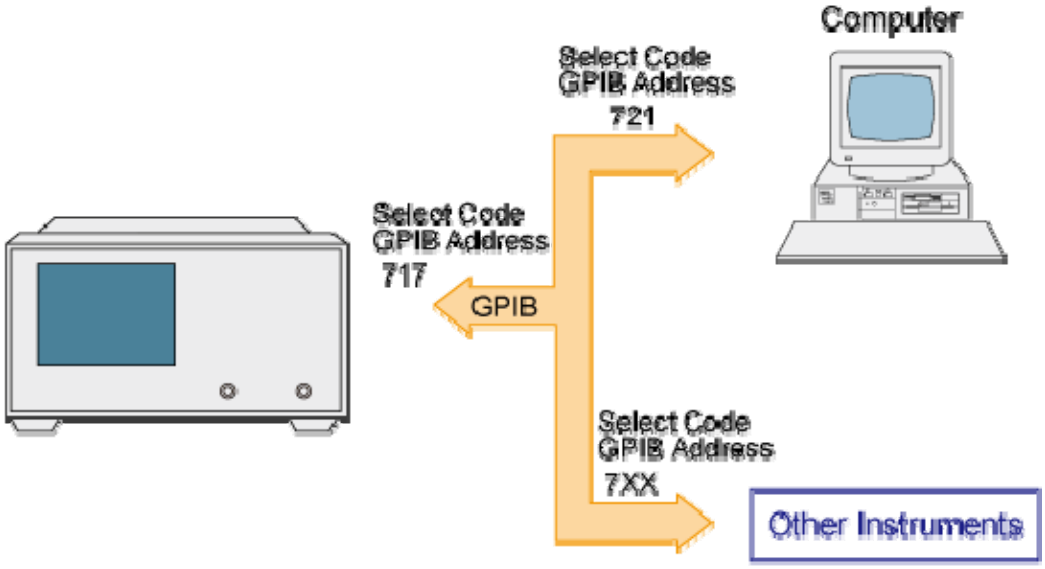
About GPIB

GPIB (General Purpose Interface Bus) is an interface standard for connecting computers and peripherals, which supports the following international standards: IEEE 488.1, IEC-625, IEEE 488.2, and JIS-C1901. The GPIB interface allows you to control the Agilent E5061B from an external computer. The computer sends commands and instructions to the E5061B and receives data sent from the E5061B via GPIB.

System Configuration

Use GPIB cables to connect between the E5061B, the external controller (computer), and peripherals. The following figure shows the overview of the system configuration of the GPIB remote control system.

Configuration of the GPIB remote control system



e5061b964

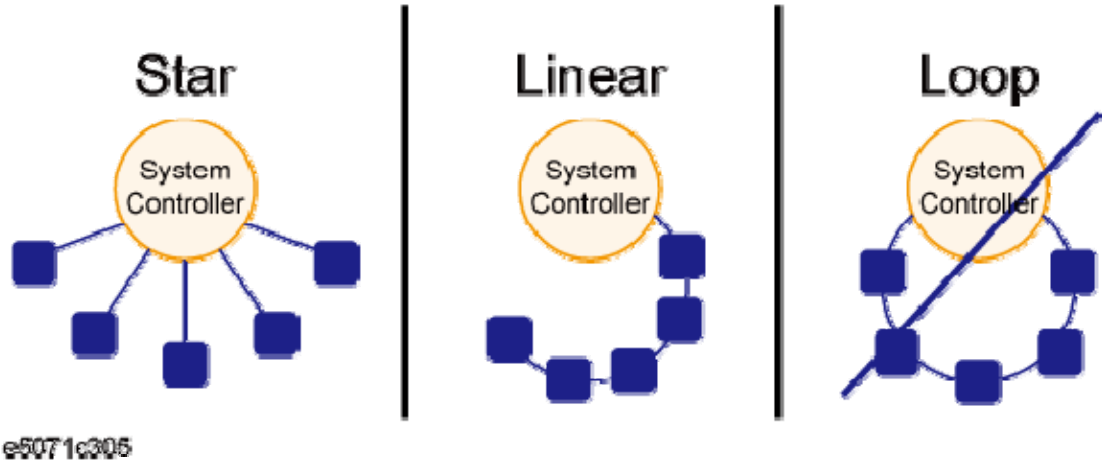
NOTE While the E5061B is turned OFF, the SRQ status of the E5061B is active. To prevent an incorrect operation on the SRQ of the GPIB remote control system, disconnect the E5061B from the system when the E5061B is turned OFF.

Required Equipment

- E5061B
- External controller (PC or workstation that can be connected to LAN and Agilent I/O Library is installed into)
- Other devices (other instruments and/or peripherals that serve your purpose)
- GPIB cables

Scale of system you can construct

- You can connect up to 15 devices in a single GPIB system.
- The length of cables to connect between devices must be 4 m or less. The total length of connecting cables in a single GPIB system must be 2 m × the number of connected devices (including the controller) or less. You cannot construct the system in which the total cable length exceeds 20 m.
- The number of connectors connected to an individual device must be 4 or less. If you connect 5 or more connectors, excessive force is applied to the connector part, which may result in failure.
- You can choose the device connection topology from star, linear, and combined. Loop connection is not supported.



Device selector

The device selector is a unique value assigned to each device that is used by the controller to select the control target (to send/receive messages) among devices connected on the GPIB remote control system.

The device selector consists of a select code (usually, 7) and a GPIB address. For example, when the select code is 7 and the GPIB address is 17, the device selector is 717. The select code must be set for each system. The GPIB address must be set to a unique value for each device,

which is used to identify devices on the same system. In the description and sample programs in this manual, it is assumed that the device selector is set to 717.

Setting the GPIB address of E5061B

To set the GPIB address for talker/listener mode, See Setting_talker_listener_GPIB_address_of_E5061B.

LAN remote control system

- Overview
- System Configuration
- Required Equipment
- Control over SICL-LAN Server
- Control using C or Visual Basic
- Control using Agilent VEE
- Control with Telnet Server
- About LXI

Other topics about Overview

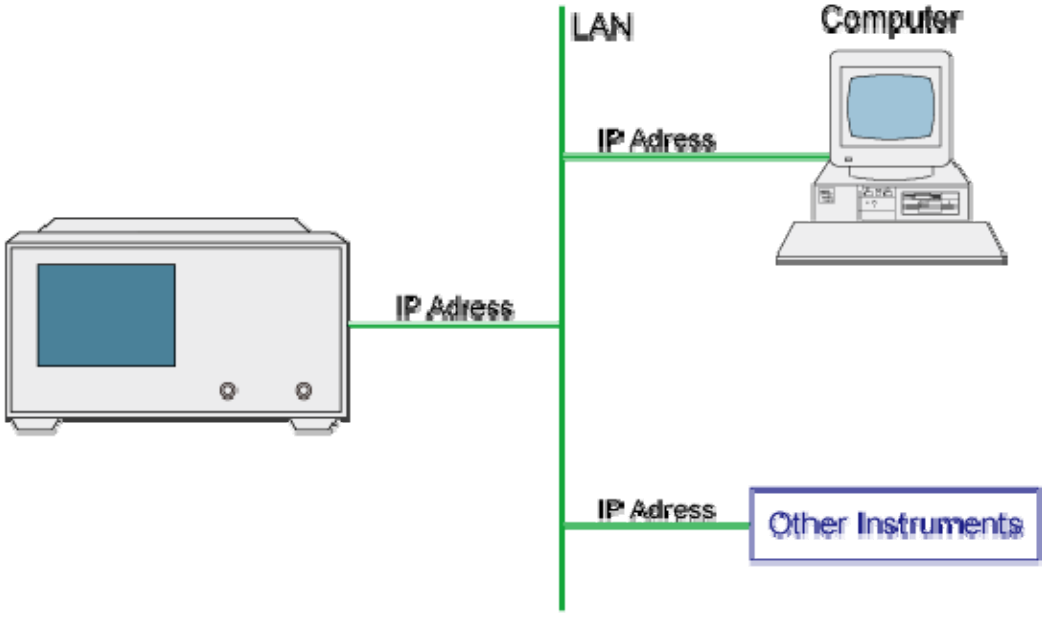
Overview

The LAN (Local Area Network) remote control system provides two methods: controlling the E5061B using the SICL-LAN server and controlling the E5061B using the telnet server.

System Configuration

Use a LAN cable to connect between the E5061B and the external controller (computer). The following figure shows the overview of the system configuration of the LAN remote control system.

Configuration of the LAN remote control system



e5061b055

Required Equipment

- E5061B
- External controller (PC or workstation that can be connected to LAN)
- Other devices (other instruments and/or peripherals that serve your purpose)
- LAN cables

Control over SICL-LAN Server

In the control system using the SICL-LAN server, communication between the external controller (client) and the E5061B (server) is performed using the SICL-LAN protocol. Communication is performed using SICL (Standard Instrument Control Library). You can control the E5061B by programming using SICL or VISA with the C language in the UNIX environment, or Visual C++, Visual Basic, or VEE in the Windows environment.

Preparing the E5061B

To communicate with the external controller, follow these steps to turn ON the SICL-LAN server of the E5061B in advance.

1. Turn ON the SICL-LAN server of the E5061B.

   System > **Misc Setup** > **Network Setup** > **SICL-LAN Server [ON]**

   NOTE When the SICL-LAN server is turned ON for the first time, the windows firewall setting dialog box appears. Select **Unblock** and click **OK**. If you select **Keep Blocking** on firewall setting, you need to unblock for the remote server in Windows firewall to use the SICL-LAN server.

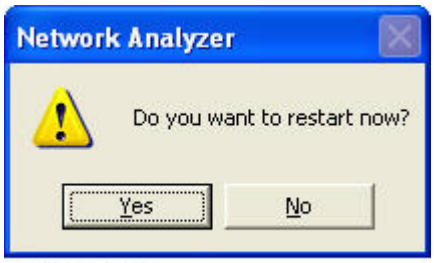
2. Set the GPIB address of the E5061B for control with the SICL-LAN server. "XX" represents an address number.

   System > **Misc Setup** > **Network Setup** > **SICL-LAN Address [XX]**

3. By default, the SICL-LAN Address does not changes until the firmware of E5061B is restarted.

e5071c137

4. On pressing any key, a message appears for restarting the firmware. Click **Yes** to restart the firmware.



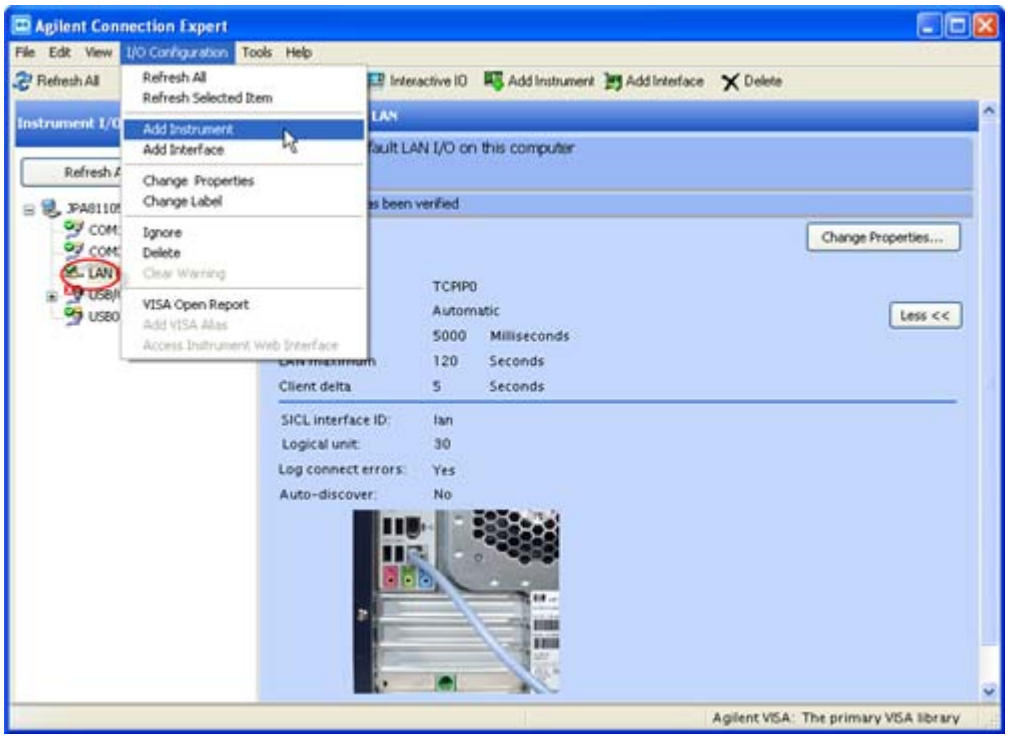
e5071c138

Preparing the external controller

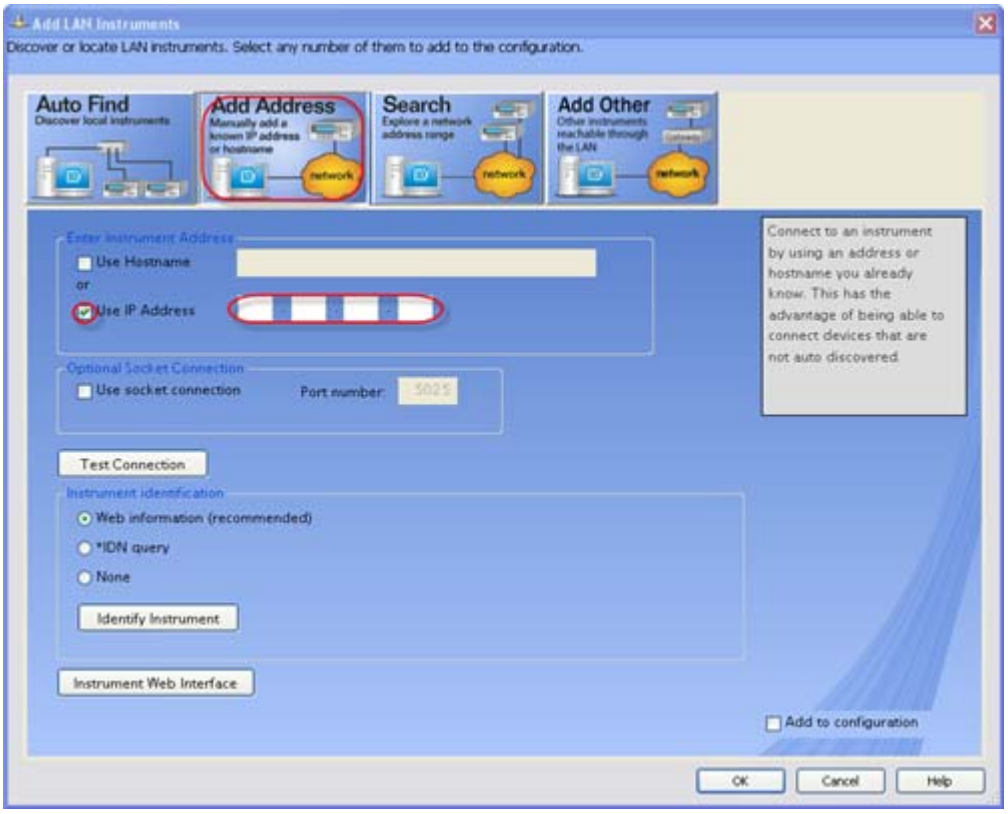
In order to establish communication with/ the E5061B using the TCP/IP protocol, you need to set the I/O interface of the external controller in advance. This section shows the setting procedure when using the external controller in the Windows environment.

NOTE You must install the Agilent I/O Libraries on your PC in advance. Use Agilent I/O Libraries Suite 14.2 or later.

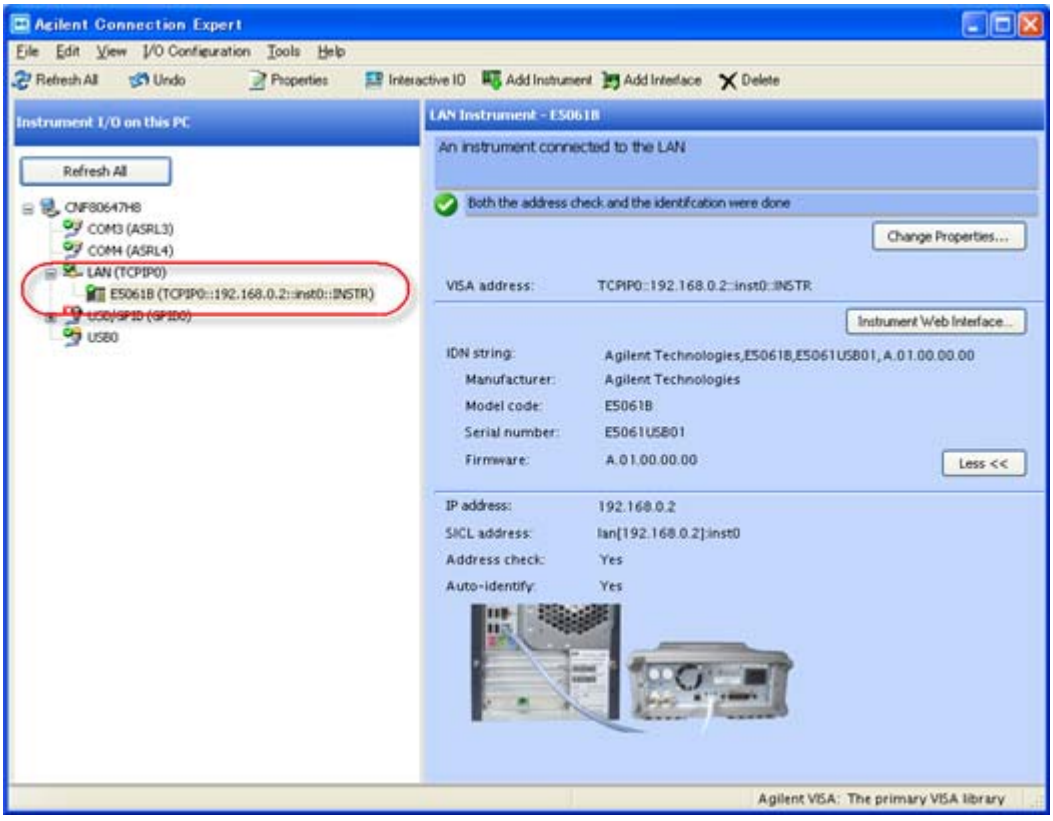
1. From your PC's Start menu, click **Program** > **Agilent I/O Libraries Suite** > **Agilent Connection Expert** to open the Agilent Connection Expert setting screen.
2. In the Agilent Connection Expert setting screen, select **LAN(TCPIP0)** in the **Instrument I/O on this PC** frame, and then click **I/O Configuration** > **Add Instrument.**



3. In the Add LAN Instrument Properties screen, set up the IP address of the E5061B and click **OK**. You can change settings as necessary. For details, refer to the Agilent I/O Libraries Suite documentation.



5. In the Agilent Connection Expert screen, check that the E5061B has been added under **LAN(TCPIP0)** in the **Instrument I/O on this PC** frame.



Control using C or Visual Basic

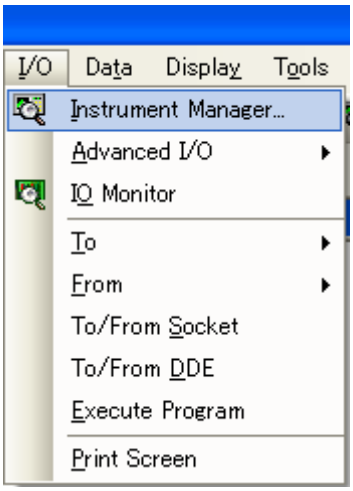
You can control the E5061B by programming using SICL with the C language in the UNIX environment, or Visual C++ or Visual Basic in the Windows environment.

Control using Agilent VEE

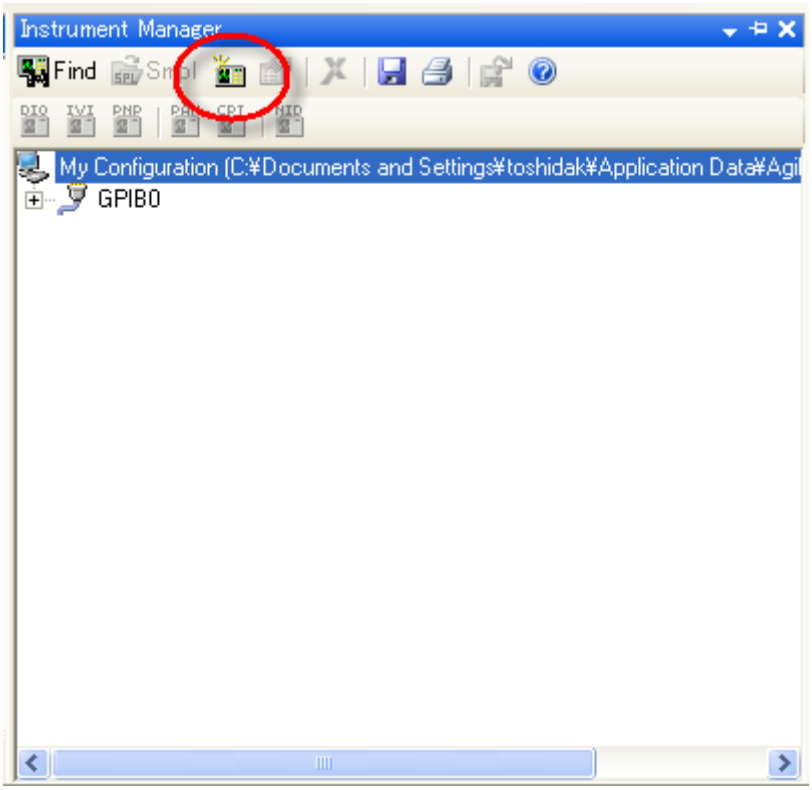
Agilent VEE allows you to control the E5061B via the I/O interface. The following example shows how to control the E5061B that is set as follows: the address of the SICL-LAN server is 17 and the IP address is 146.208.116.90.

NOTE When using Agilent VEE for PC, use Agilent VEE Pro 7.5 for Windows or later.

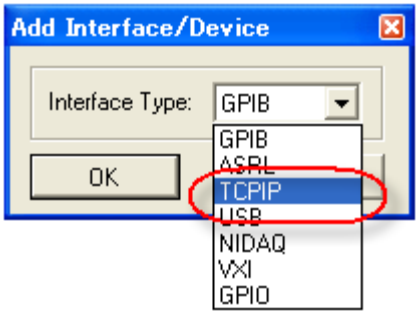
1. On the Agilent VEE's **I/O** menu, click **Instrument Manager**



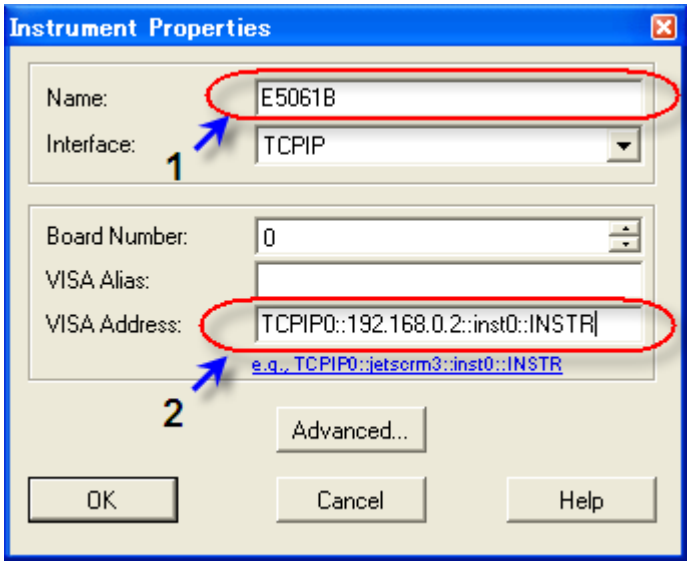
2. In **Instrument Manager**, click **Add Instrument** Icon.



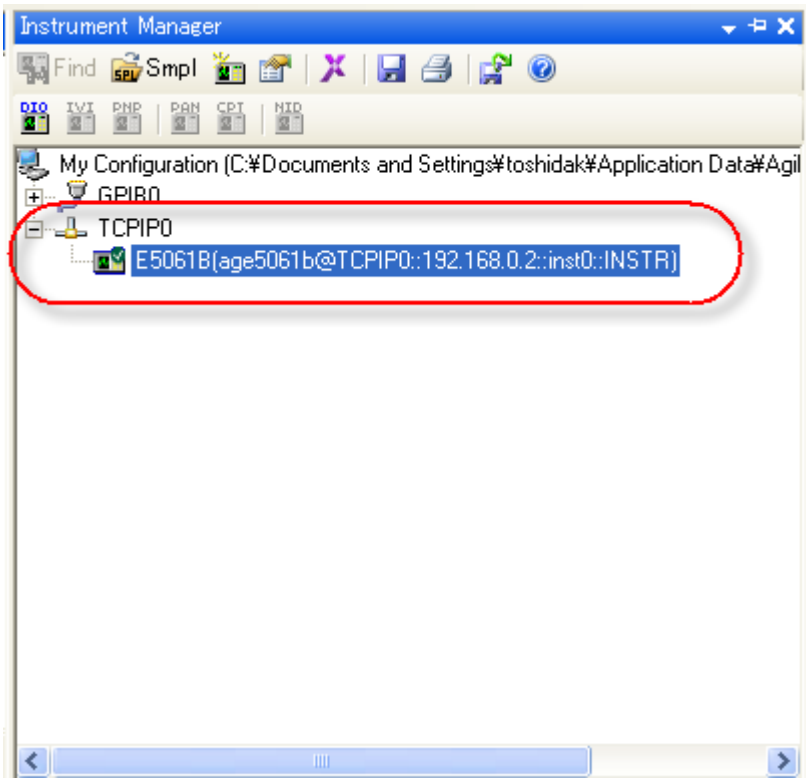
3. A new windows appears for the selection of Interface Type. Select **TCPIP** and click **OK**.



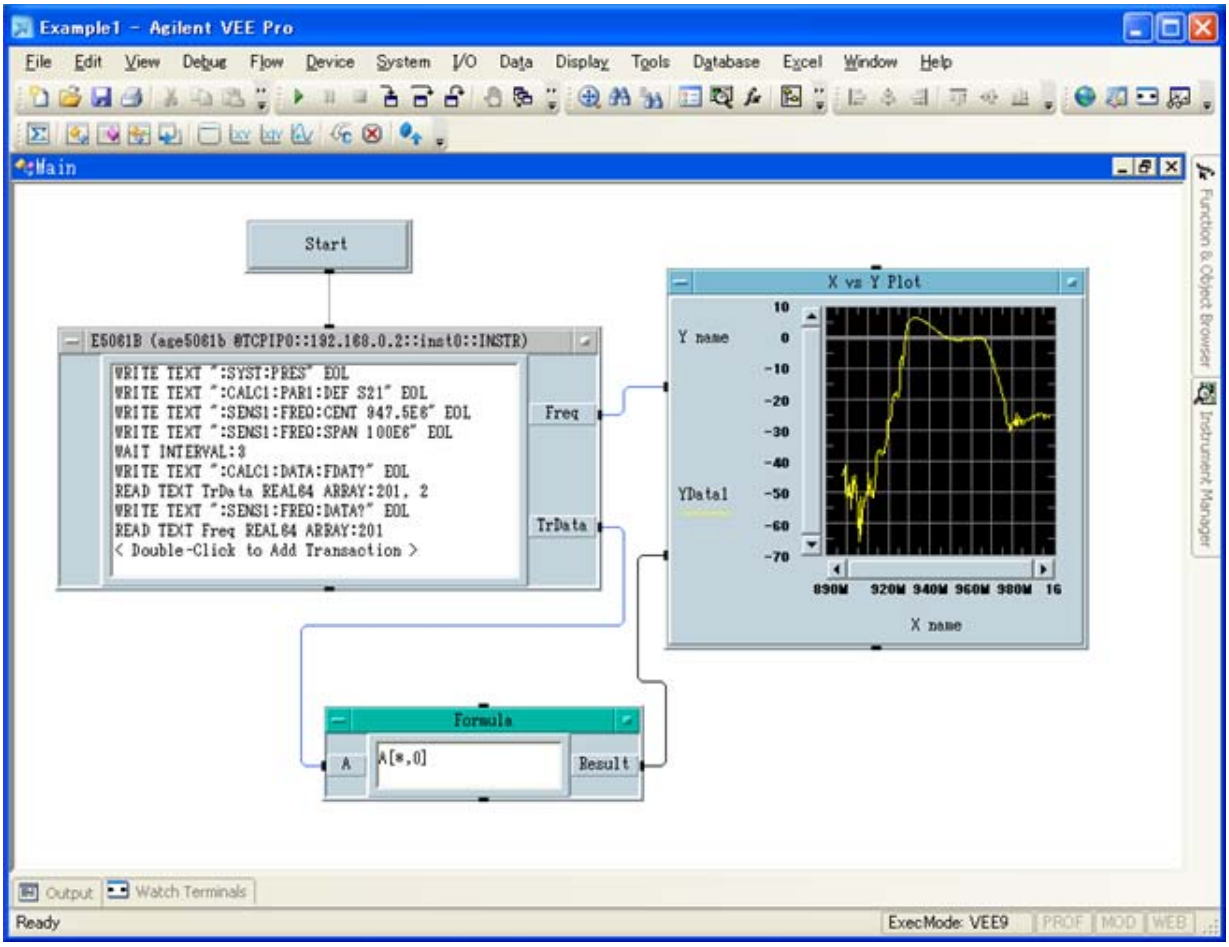
4. In **Instrument Properties** dialog box, type any name for the Instrument in Name (1 in the following figure, for example: ENA or E5061B), and add TCPIP0::<IP Address>::inst0::INSTR in the **TCIP Address,** where <IP Address> is the IP address for E5061B (2 in the following figure). For example, if the IP address for E5061B is 192.168.0.2, then the value for **TCPIP Address** would be TCPIP0::192.168.0.2::inst0::INSTR. Click **OK** after entering all the parameters.



5. The Instrument manager displays the connection with E5061B.



The following figure shows an example of control using the I/O interface that has been set in the above procedure.



Control with Telnet Server

In the control system over telnet server, communications are performed through connection between the sockets provided by the processes of the external controller and the E5061B to establish a network path between them.

A socket is an endpoint for network connection; port 5024 and port 5025 are provided for the sockets for the E5061B. Port 5024 is provided for conversational control using telnet (user interface program for the TELNET protocol) and port 5025 for control from a program.

NOTE To use telnet, port 5024 and 5025 should be opened through Windows firewall.

Preparing the E5061B

To communicate with the external controller, follow these steps to turn on the telnet server of the E5061B in advance.

System > **Misc Setup** > **Network Setup** > **Telnet Server [ON]**

NOTE When the telnet server is turned ON for the first time, the windows firewall setting dialog box appears. Select

Unblock and click **OK**. If you select **Keep Blocking** on firewall setting, you need to unblock for the remote server in Windows firewall to use the telnet server.

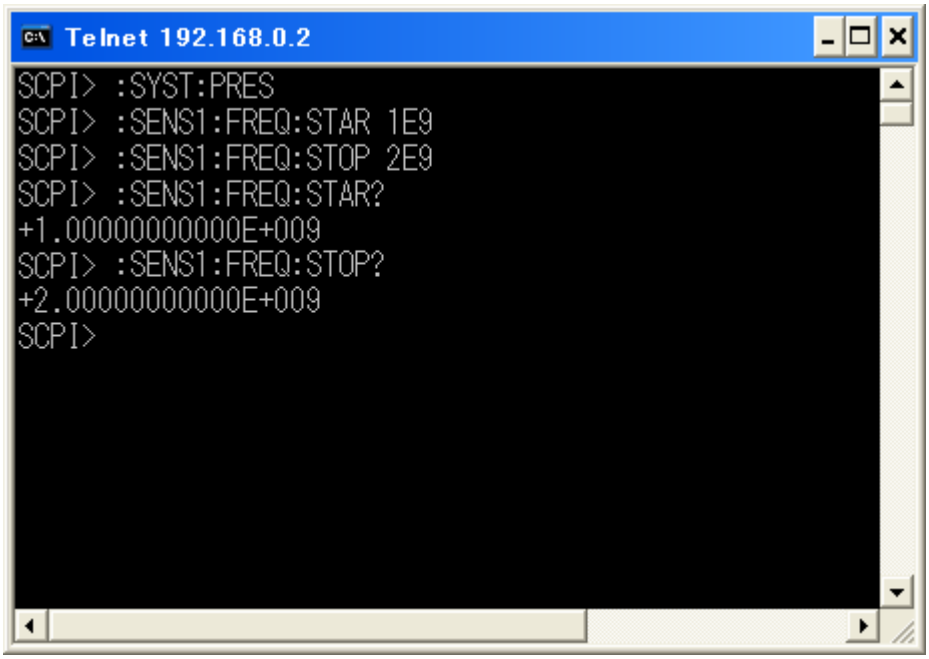
Conversational control using telnet (using port 5024)

You can use telnet to perform conversational control by sending SCPI commands to the E5061B on a message-by-message basis. For telnet, the socket of port 5024 is used for communications.

In this example, in order to show you the control procedure using telnet, you control the E5061B (IP address: 192.168.0.2 and host name: e5061b) from the external controller in the Windows environment.

1. Open the MS-DOS command prompt screen.
2. At the MS-DOS prompt, type **telnet 192.168.0.2 5024** and press the return key.
3. The telnet screen opens.
4. Type a command and press the return key; it is sent to the E5061B and executed. If you enter a command that queries some data, the query response is displayed below the line you have entered the command.
5. The following figure shows the screen after using the :SYST:PRES command to reset, the :SENS{1-4}:FREQ:STAR command and :SENS{1-4}:FREQ:STOP commands to set the sweep start value and stop value to 1 GHz and 2 GHz respectively, and checking the settings.

Example of control using telnet



6. Press] while holding down Ctl in the telnet screen to break the connection with the E5061B. The telnet prompt appears. At the telnet prompt, type quit and press the Enter key. The connection to the E5061B breaks and telnet ends.

Control from a program (using port 5025)

When controlling the E5061B from a program on the external controller, use the socket of port 5025 for connection.

NOTE Some functions such as service requests that are available in the GPIB remote control system are not available in control over telnet server.

Control using C or Visual Basic

You can control the E5061B by socket programming using the C language in the UNIX environment, or Visual C++ or Visual Basic in the Windows environment.

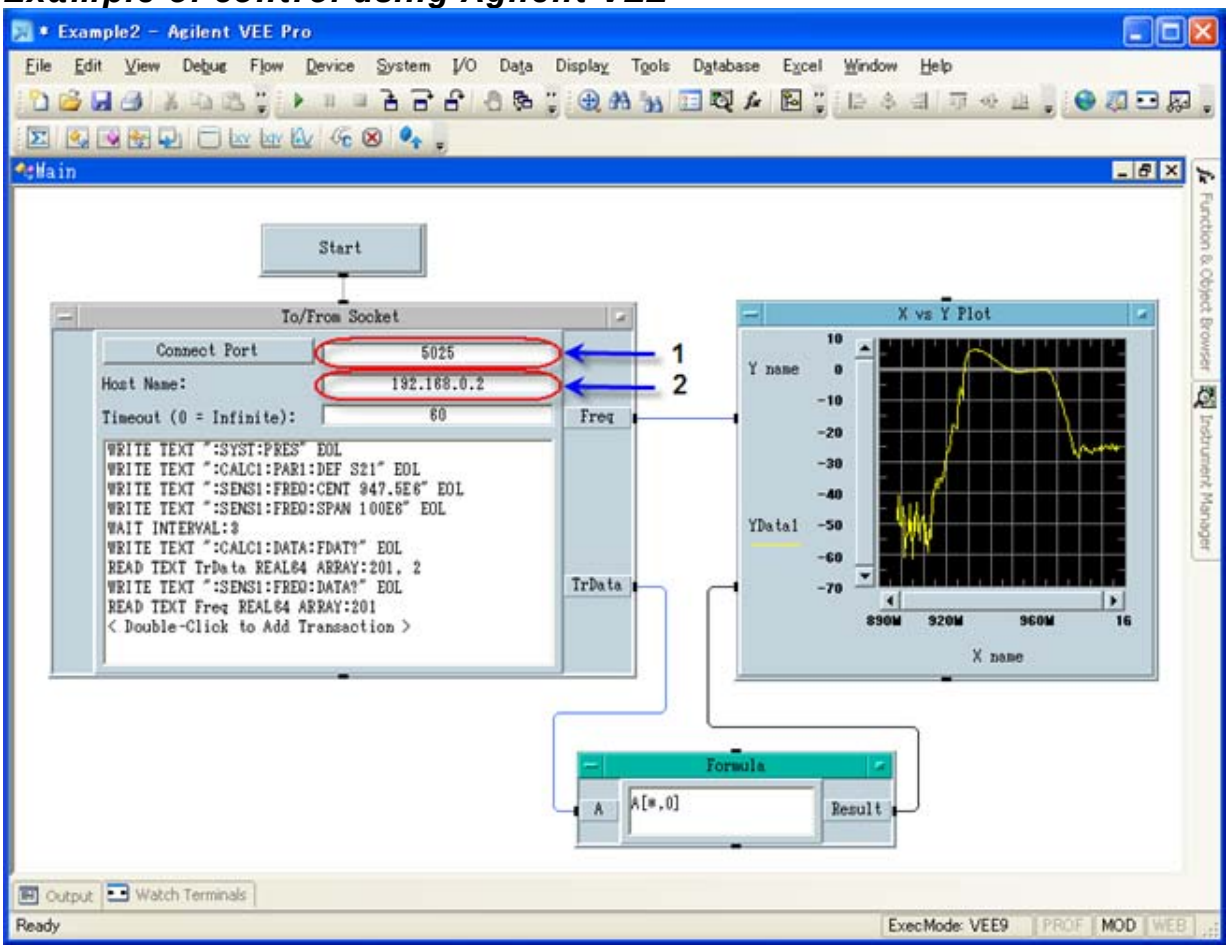
For socket programming, the library for network connection on the TCP/IP protocol is required. For the UNIX environment, BSD (Berkeley Software Distribution) Sockets API is available; for the Windows environment, WinSock (WinSock1.1 and WinSock2.0) is created by porting BSD Sockets to Windows and expanding is available.

For more information on the control method, see a sample program for control using WinSock described in "Controlling Using Telnet Server".

Control using Agilent VEE

Agilent VEE allows you to control the E5061B through the connection to the socket of port 5025 using To/From Socket. The following figure shows an example (when the IP address of the E5061B is 192.168.0.2). Enter 5025 in **Connect Port** to specify the port for connection (1 in the following figure) and enter the IP address or host name of the E5061B in the field to specify the **Host Name** (2 in the following figure).

Example of control using Agilent VEE



About LXI

LXI (LAN eXtensions for Instrumentation) is the LAN-based successor to GPIB and combines the advantages of Ethernet with the simplicity and familiarity of GPIB. The key features of LXI are as follows:

- The speed, simplicity, worldwide reach, low cost, ongoing enhancement and backward compatibility of LAN.
- Quick, easy configuration through the intuitive web interface built into compliant instruments.
- Simplified programming and greater software reuse through IVI drivers.

- The ability to create hybrid systems that include LXI, GPIB, VXI, PXI, CANbus, etc.
- Enhanced system performance and event handling via hardware- and LAN-based triggering modes.
- Synchronization of local and remote instruments through the IEEE 1588 precision time protocol.

NOTE For more information on LXI, refer to www.lxistandard.org

USB Remote Control System

- Overview
- System Configuration

Other topics about Overview

Overview

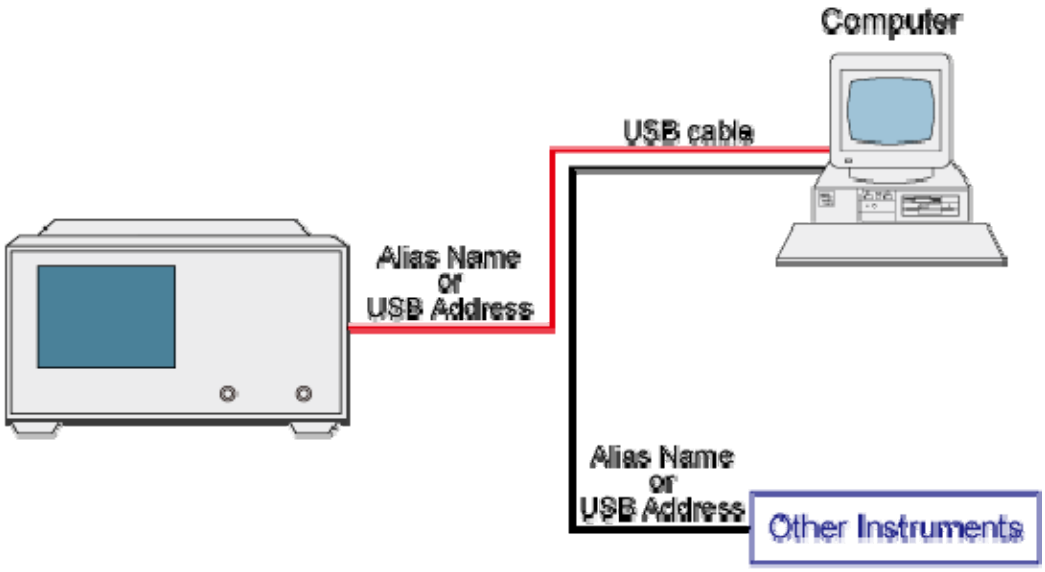
The USB (Universal Serial Bus) remote control system provides device control via USB, which is equivalent to control via GPIB. Connection is made through an interface in compliance with USBTMC-USB488 and USB 2.0.

System Configuration

The USB remote control system controls instrument with either the name "alias" or the USB address.

Use a USB cable to connect the E5061B to an external controller (personal computer). The following figure shows an overview of the system configuration for the USB remote control system.

USB Remote Control System Configuration



e5061b003

Required Equipment

- E5061B
- External controller (PC with USB host port (type A)).

- Other USB compatible devices (instruments and/or peripherals for specific purposes).
- USB cable connecting E5061B and external controller (with type A/4-prong male or type B/4-prong male connectors depending on device used).

USB Port Types

There are two standard types of USB ports. The external controller (PC) must be connected via the USB host port (type A), while the E5061B and other USB compatible devices must be connected via the USB interface port (type B).

Port Type	Description
	Type A: USB host port
	Type B: USB (USBTMC) interface port

Preparing E5061B

You do not have to configure any softkey or command of the E5061B in order to control the E5061B from an external controller. Simply connect a USB cable to the USB interface port.

Preparing External Controller

In order to establish communication with the E5061B via USB, you must set up the I/O interface of the external controller in advance. The USB can identify devices automatically, so once you connect a USB cable to a target device, a dialog box appears for USB device registration.

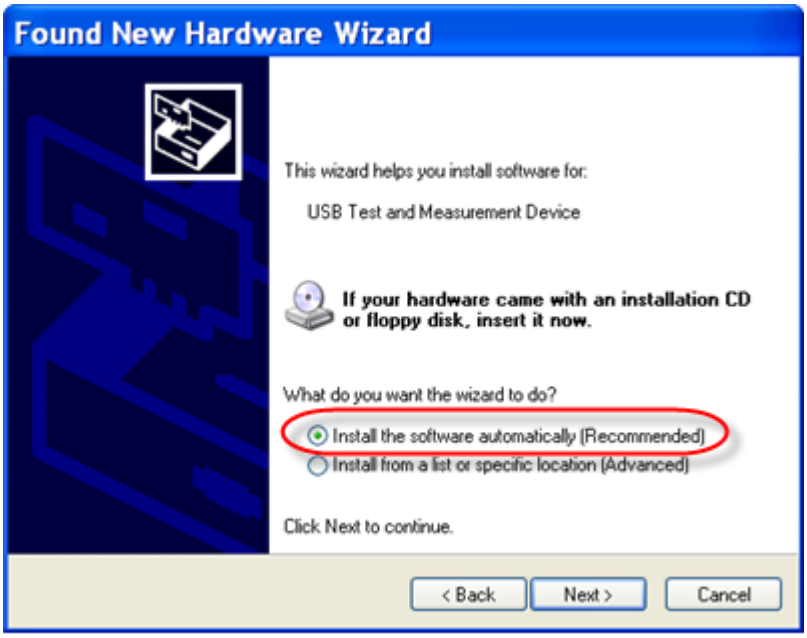
NOTE The E5061B is identified as new device if its serial number has been changed.

NOTE You must install the Agilent I/O Libraries on your PC in advance. Use Agilent I/O Libraries Suite 14.2 or later.

1. Setting E5061B when USB Cable Is Connected
    1. When new device is connected via USB cable, the following dialog box appears automatically. Select **No, not this time**, and then click **Next >**.

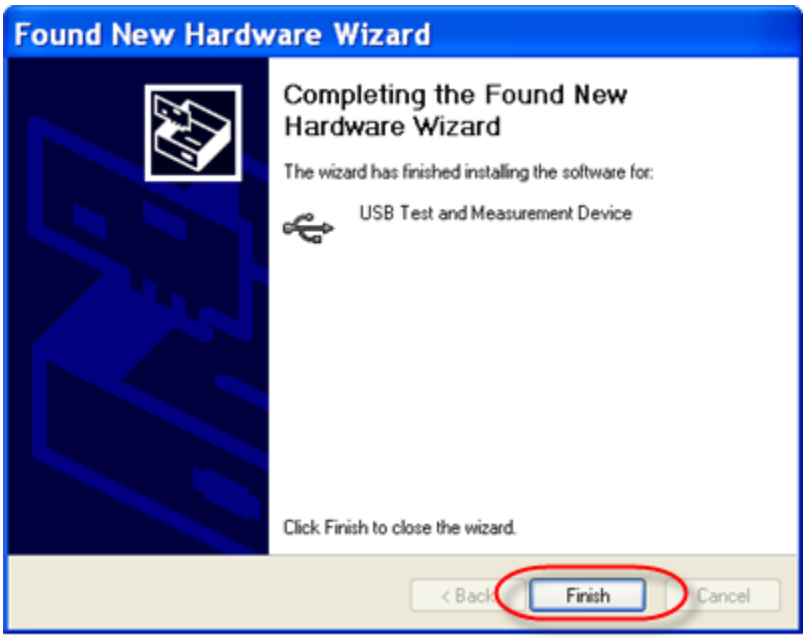
e5071c155

2. Select **Install the software automatically (Recommended)**, and then click **Next >**.



e5071c156

3. The drivers for E5061B are automatically installed and the completion screen appears. Click **Finish** to complete the process.



e5071c157

4. If you use Agilent I/O libraries 14.x and below, after finishing the setting, the dialog box named "Assign USB device alias" appears. Check "When a new USB device is plugged in.", press OK.

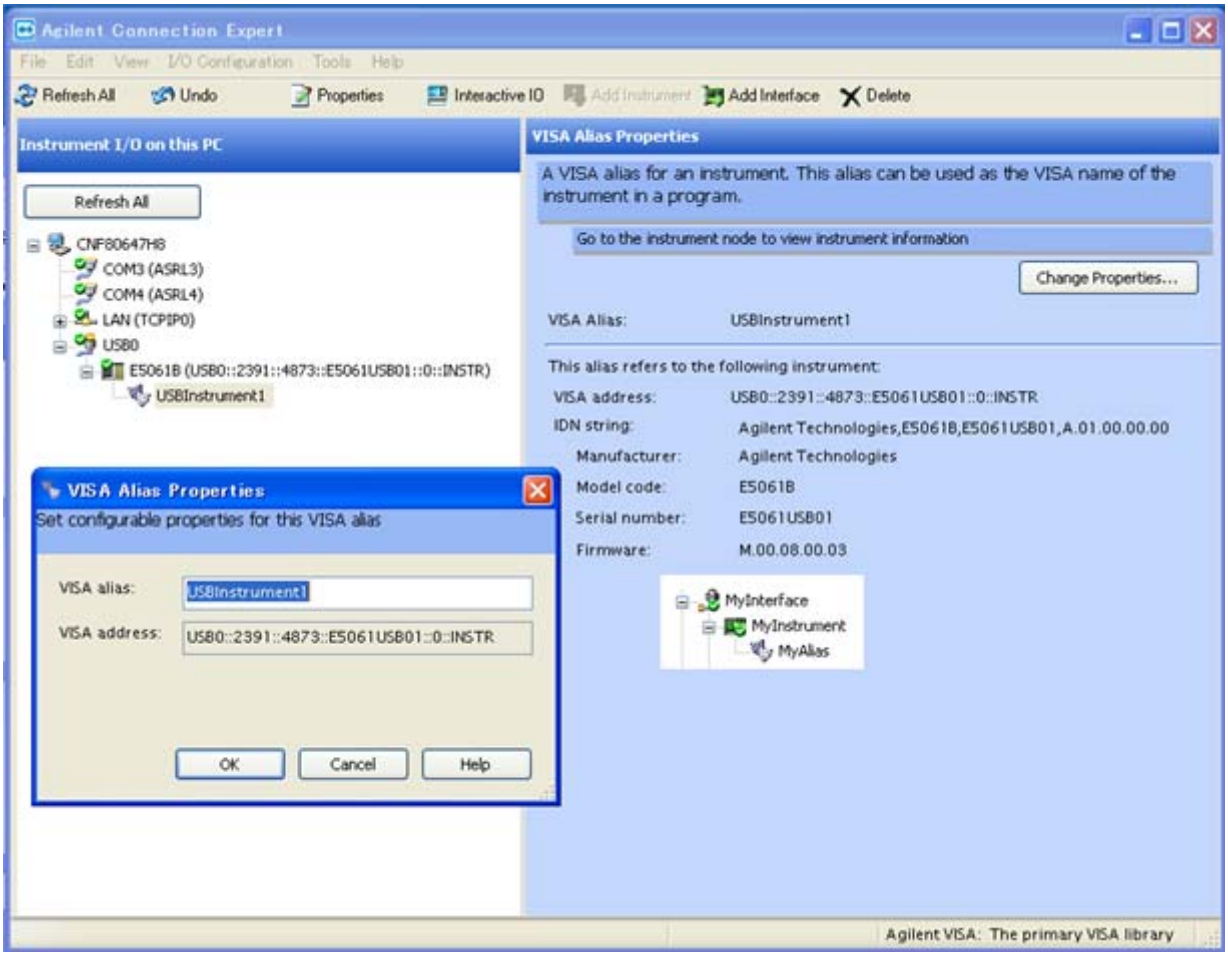
   NOTE When the device is detected at the different USB port, the "New Hardware Search Wizard" will start. Follow the instruction to implement the processing.

3. Changing Alias on Setting Screen

The following are steps using the Agilent I/O Libraries Suite 15.

1. From the Start menu of your PC, click **Programs** > **Agilent IO Libraries Suite** > **Agilent Connection Expert** to open the Config setting screen.
2. In the Config setting screen, select the alias names from **USB0** onward in the **Instrument I/O on this PC** frame, and then use the **Change Properties** from **I/O Configuration** on the menu bar.

Changing Alias



NOTE For alias, use the ASCII format less than 127 digits. Alias is upper/lower case insensitive.

Control using C or Visual Basic

You can control the E5061B by programming using Visual C++ or Visual Basic in the Windows environment as well as SICL/VISA.For further information on controlling the E5061B, see the manual of SICL or VISA. Use Agilent I/O Libraries Suite 14.2 or later.

You may use alias in the programming using SICL/VISA.

The following example shows an OPEN command to control the E5061B to which alias is given as ENA_USBIF.

SICL	id = iopen("ENA_USBIF")
VISA	viOpen(...,"ENA_USBIF",...)

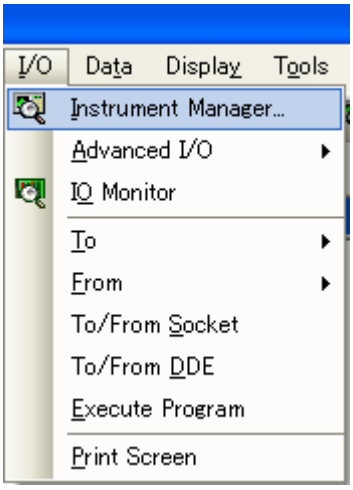
NOTE For further details of the programming using SICL/VISA, see the SICL Users Guide or the VISA Users Guide.

Control using Agilent VEE

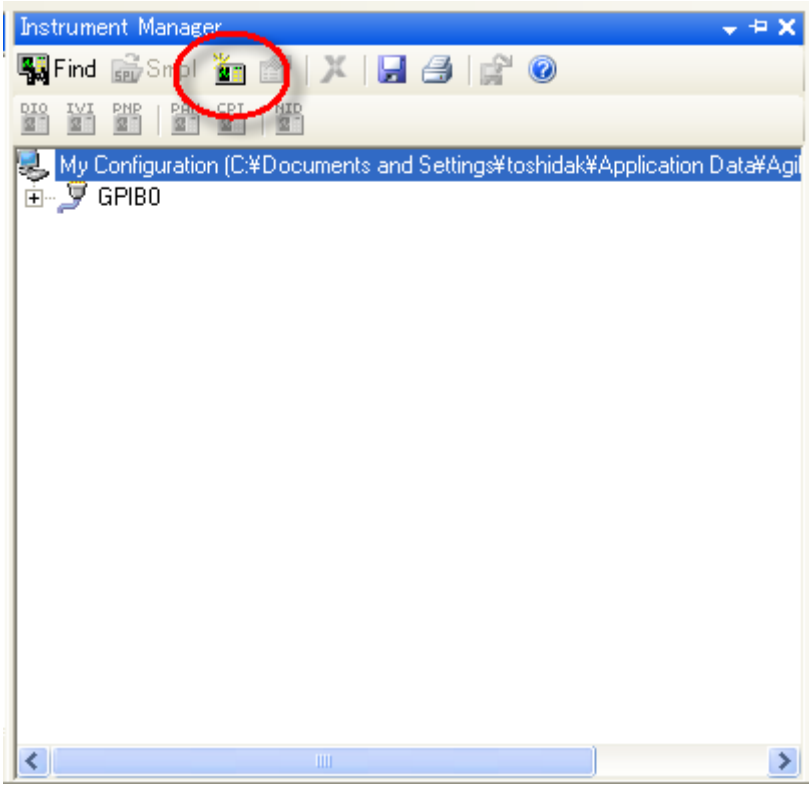
Agilent VEE allows you to control the E5061B via the direct I/O interface. The following example shows how to control the E5061B to which alias is given as ENA_USBIF.

NOTE When using Agilent VEE for PC, use Agilent VEE Pro 7 for Windows or later version.

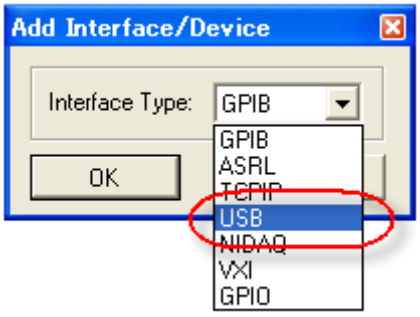
1. On the Agilent VEE's **I/O** menu, click **Instrument Manager**.



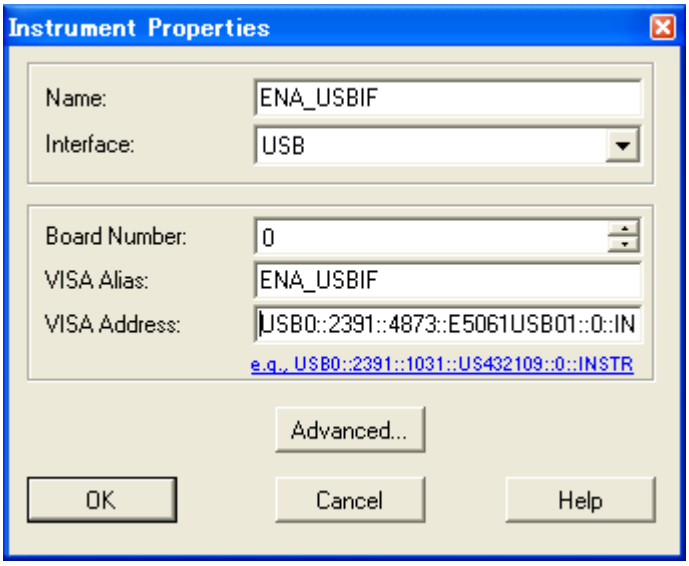
2. In **Instrument Manager**, click **Add Instrument** Icon.



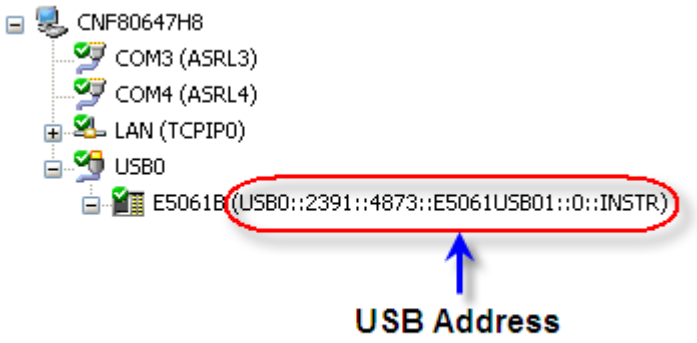
3. A new windows appears for the selection of Interface Type. Select **USB** and click **OK**.



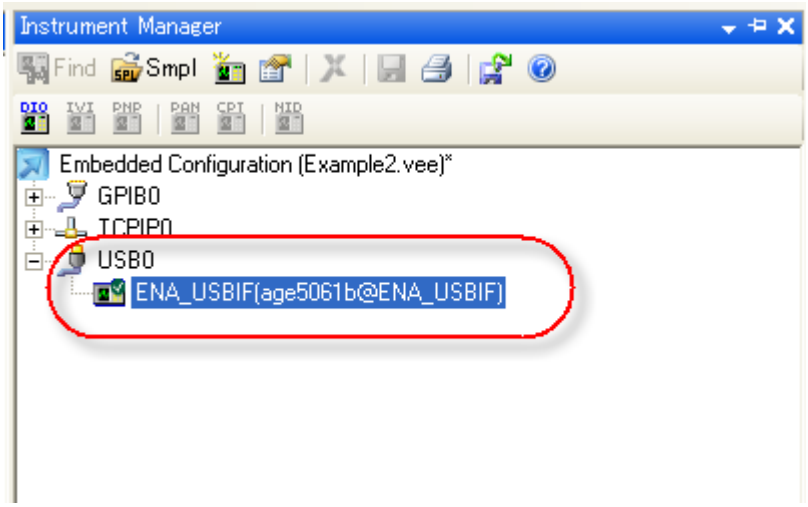
4. In **Instrument Properties**, type any name for the Instrument in Name (for example: ENA_USBIF or E5061B_USB), and add USB Address in the **VISA Address.** Click **OK** after entering all the parameters.



NOTE The USB address can be retrieved from Agilent Connection Expert.



5. The E5061B successfully appears in the **Instrument Manager**.



Sending SCPI command messages

- Type and Structure of Commands
- Grammar of Messages
- Remote Mode

Other topics about Overview

Type and Structure of Commands

The SCPI commands available for the E5061B are classified into 2 groups as follows.

E5061B commands

Commands specific to the E5061B. They cover all measurement functions that the E5061B has and some general-purpose functions. The commands in this group are arranged in a hierarchical structure called the command tree. Each command consists of character strings (mnemonics) indicating each hierarchical level and colon (:) separators between hierarchical levels.

IEEE common commands

Commands to cover general-purpose functions defined in IEEE488.2 that are available commonly to instruments that support this standard. The commands in this group have an asterisk (*) at the beginning. For the commands in this group, there is no hierarchical structure.

Concepts of the command tree

The commands at the top of the command tree are called "root command" or simply "root." To access lower level commands in the tree, you need to specify a specific path like a directory path in the DOS file system. After power-on or reset, the current path is set to the root. Special characters in messages change the path setting as described below.

Message terminator

A message terminator such as the <new line> character sets the current path to the root.

Colon (:)

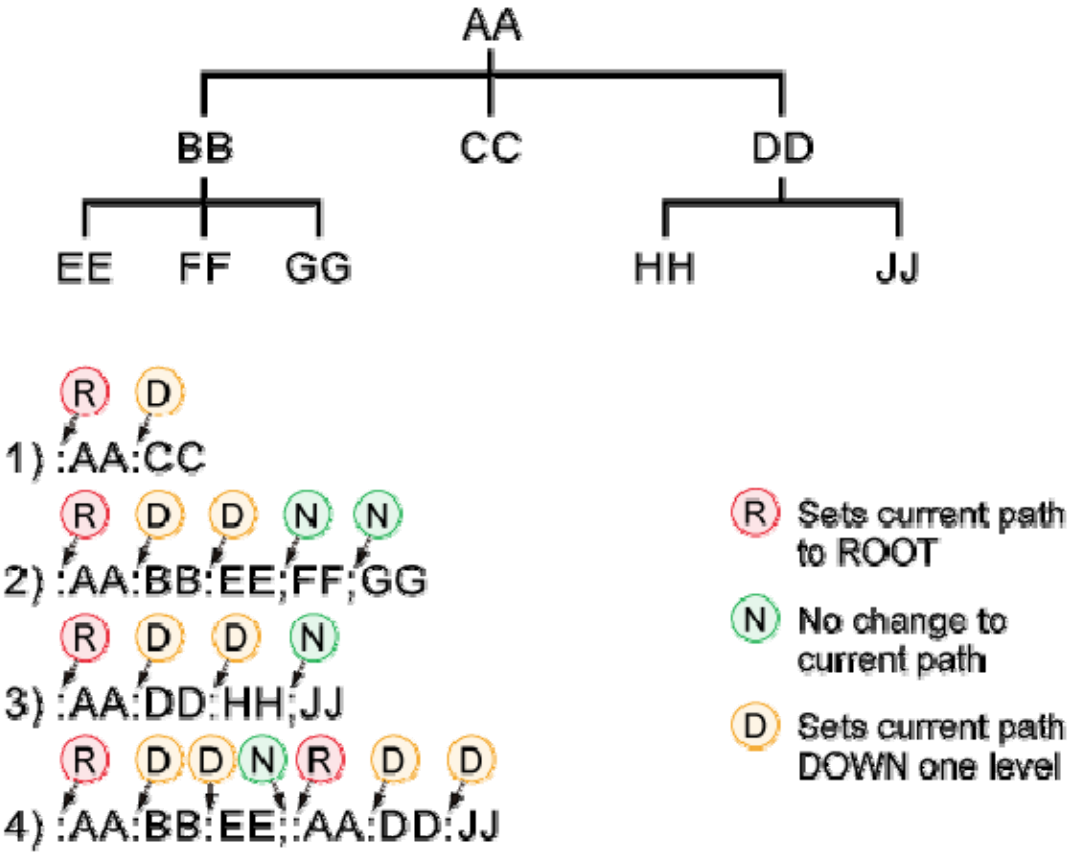
A colon between 2 command mnemonics lowers the level of the current path in the command tree. A colon used as the first character of a command specifies the command mnemonic that follows as the root-level command.

Semicolon (;)

A semicolon does not change the current path and separates 2 commands in the same message.

The following figure shows an example of how to use colons and semicolons to efficiently access commands in the command tree.

Using colons and semicolons



e5071c350

Grammar of Messages

This section describes the grammar to send program messages via GPIB. Program messages are messages that the user sends to the instrument from the external controller to control the instrument. A program message consists of 1 or more commands and their necessary parameters.

Upper/lower case sensitivity

Upper/lower case insensitive.

Program message terminator

A program message must be terminated with one of the 3 program message terminators: <new line>, <^END>, or <new line><^END>.

<^END> indicates that EOI on the GPIB interface becomes active at the instant when the immediately previous data byte is sent. For example, the OUTPUT command of HTBasic automatically sends the message terminator after the last data byte.

Parameters

A space (ASCII code: 32) is required between a command and its first parameter. When sending several parameters in a single command, separate each parameter with a comma (,).

Message including several commands

When sending 2 or more commands in a single message, separate each command with a semicolon (;). The following example shows how to send the *CLS command and the :STAT:PRES command in a single message using HTBasic.

```
OUTPUT 717;"*CLS;:STAT:PRES"
```

Remote Mode

The E5061B does not provide remote mode. Therefore, even if you send a GPIB command, it never enters into the remote mode automatically. There is no local key to release remote mode.

If you need to prevent misoperation during remote control due to entry from the front panel or mouse, lock the input devices using the following commands.

- :SYST:KLOC:KBD
- :SYST:KLOC:MOUS

Setting up Analyzer

Setting up Analyzer

- Selecting the Active Channel/Trace
- Configuring Measurement Conditions
- Configuring Display Settings
- Saving and Loading the Settings

Selecting the Active Channel / Trace

You can configure the E5061B by using various commands. Some commands require you to specify and work with a particular channel or trace, while other commands do not have this restriction.

Those commands that do not require you to specify a particular channel or trace apply to the currently active channels and traces. Before issuing such a command, therefore, you must make the appropriate channels and traces active.

To make a channel active, use the following command:

:DISP:WIND{1-4}:ACT

NOTE Only the currently displayed channels can be active channels. Therefore, you must display the desired channels by using the :DISP:SPL command before making them active.

To make a trace active, use the following command:

:CALC{1-4}:PAR{1-4}:SEL

NOTE Only the currently displayed traces can be active traces. Therefore, you must display the desired traces by using the :CALC{1-4}:PAR:COUN command before making them active.

Other topics about Setting up Analyzer

Configuring Measurement Conditions

Other topics about Setting up Analyzer

Setting the Number of Traces

When you set the number of traces, that setting determines the upper limit trace number; for example, if the setting is 3, traces 1 through 3 are displayed. To set the number of traces, use the following command:

:CALC{1-4}:PAR:COUN

NOTE Only the currently displayed traces can be active traces. Therefore, you must set the number of traces appropriately before making them active.

Selecting Measurement Parameters

To select the measurement parameter (S parameter) for each trace, use the following command:

:CALC{1-4}:PAR{1-4}:DEF

There is no equivalent command with **Meas** > **Measurement Port**. Using this command allows you to select the parameters for Gain-Phase and S-Parameter measurements in one channel. (For example, Trace 1: T/R, Trace 2: S11).

Setting Sweep Condition (Stimulus)

To select one of the above sweep types, use the following command:

:SENS{1-4}:SWE:TYPE

Turning ON/OFF stimulus signal output

To turn ON/OFF the stimulus signal output, use the following command:

:OUTP

Configuring linear/log sweep settings

To set the sweep range, use the following commands:

Type	Command
Start value	:SENS{1-4}:FREQ:STAR

Stop value	:SENS{1-4}:FREQ:STOP
Center value	:SENS{1-4}:FREQ:CENT
Span value	:SENS{1-4}:FREQ:SPAN

To set the number of measurement points, use the following command:

:SENS{1-4}:SWE:POIN

To set the sweep time, use the following commands:

Type	Command
Sweep time	:SENS{1-4}:SWE:TIME
Turning ON/OFF auto setting	:SENS{1-4}:SWE:TIME:AUTO

To set the sweep delay time, use the following command:

:SENS{1-4}:SWE:DEL

To set the IF bandwidth, use the one of the following commands (both provide the same function):

:SENS{1-4}:BAND

:SENS{1-4}:BWID

To set the AUTO IF bandwidth, use the following command:

:SENS{1-4}:BWA

Setting power level

To set the power level, use the following command:

:SOUR{1-4}:POW

To select whether to output the same power level (the set value for port 1) or a different power level for each port, use the following command:

:SOUR{1-4}:POW:PORT:COUP

:SOUR{1-4}:POW:PORT{1-2}

Configuring segment sweep settings

When you opt to use segment sweep, you can set all items (in the segment sweep table) by using a single command:

:SENS{1-4}:SEGM:DATA

Alternatively, you can configure the segment sweep settings based on the data contained in a CSV file by issuing the following command:

:MMEM:LOAD:SEGM

Also, you can save the contents of the current segment sweep table to a file by issuing the following command:

:MMEM:STOR:SEGM

For more information on how to save and load the segment sweep table, refer to Saving and recalling the segment sweep table.

Configuring power sweep settings

To set the sweep range, use the following commands:

Type	Command
Start value	:SOUR{1-4}:POW:STAR
Stop value	:SOUR{1-4}:POW:STOP
Center value	:SOUR{1-4}:POW:CENT
Span value	:SOUR{1-4}:POW:SPAN

To set the fixed frequency (CW frequency), use the following command:

:SENS{1-4}:FREQ

To set the number of points, the sweep time, the sweep delay time, and the IF bandwidth, use the same commands as for the linear/log sweep.

Configuring Averaging Settings

To configure the averaging settings, use the following commands:

Type	Command
On/off	:SENS{1-4}:AVER
Averaging factor	:SENS{1-4}:AVER:COUN
Clear (Restart)	:SENS{1-4}:AVER:CLE

For averaging, normally, the instrument must be triggered according to the number of averaging; however, when the averaging trigger is turned ON, sweeps for the number of averaging can be executed by a single trigger. For details on the averaging trigger, refer to Averaging Trigger Function.

Setting the System Z_0

To set the system characteristic impedance (Z_0), use the following command:

:SENS:CORR:IMP

Configuring Display Settings

- Setting the Layout of Windows and Graphs
- Configuring Trace Display Settings
- Setting Display Color

Other topics about Setting up Analyzer

Setting the Layout of Windows and Graphs

You can split the E5061B's LCD screen into multiple windows that display channel-specific result information, and the window layout can be selected from a number of variations. In addition, you can place on screen a segment sweep table or echo window, which you can use to display messages from your custom program.

Selecting the window layout (Channel Display Mode)

One window displays the results for a single channel. You cannot have a single window to display the results from more than one channel. This means that setting the window layout determines the number of channels displayed on screen.

To select one of the window layouts shown in the figure below, use the following command:

:DISP:SPL

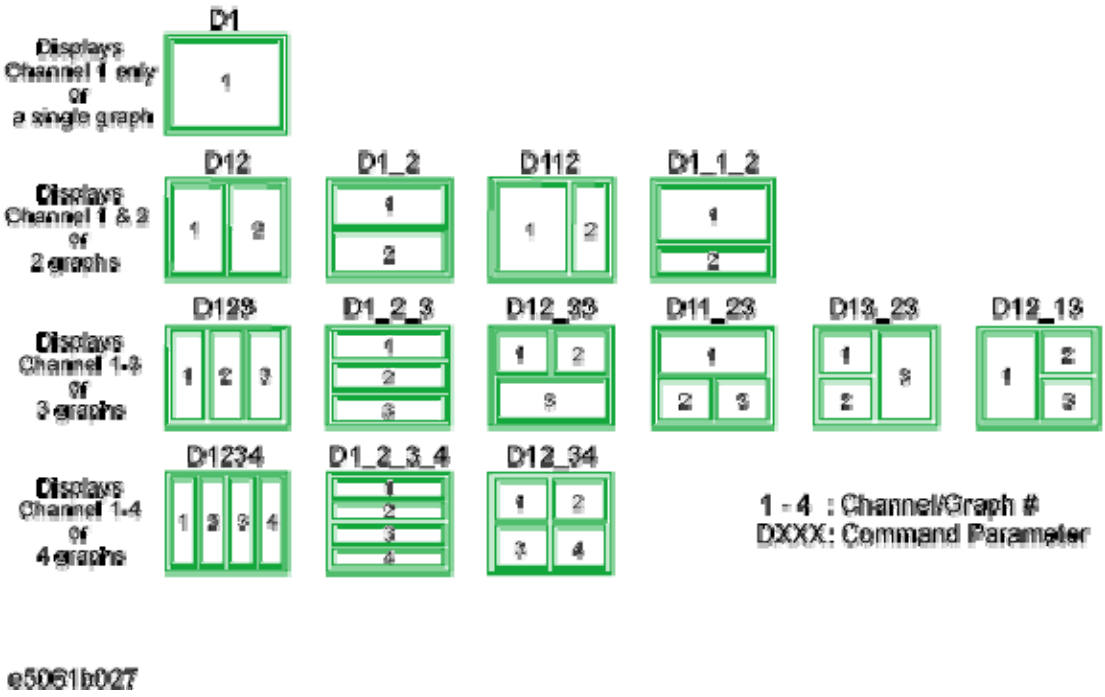
Selecting the graph layout (Trace Display Mode)

You can place a number of trace graphs in each window by selecting one of the pre-defined graph layouts. The number of graphs differs depending on your selected graph layout. If the number of graphs is equal to or larger than the number of traces (set by the :CALC{1-4}:PAR:COUN command), each graph always displays one trace. On the other hand, if the number of graphs is smaller than the number of traces, some of the graphs display two or more traces. Graph 1 is populated with trace 1, graph 2 with trace 2, and so on. Traces whose numbers exceed the last graph's number populates graph 1, graph 2, and so on.

To select one of the 14 different graph layouts shown in the figure below, use the following command:

:DISP:WIND{1-4}:SPL

Window/graph layouts and command parameters



Maximizing a window or a trace graph

When you have multiple windows displayed, you can maximize the active channel window so that it covers the entire screen area. To maximize a window, use the following command:

:DISP:MAX

Similarly, when you have multiple traces displayed, you can maximize the active trace so that it extends throughout the entire window. To maximize a trace, use the following command:

:DISP:WIND{1-4}:MAX

Showing/hiding a table or echo window

You can display the following items at the bottom of the LCD screen:

- Segment sweep table
- Limit table
- Marker list table
- Echo window (a window that displays messages from a custom program)
- Loss compensation table

To show or hide each of the above items, use the following command:

:DISP:TABL

You cannot have two or more of the above items displayed at a time. The screen displays only the selected item by using the following command:

:DISP:TABL:TYPE

Showing/hiding softkey labels

You can show or hide the softkey labels placed alongside the right-hand edge of the LCD screen. To show or hide the softkey labels, use the following command:

:DISP:SKEY

Configuring Trace Display Settings

Selecting which traces to display

Each trace has two different representations: data and memory traces. You can show or hide the data and memory traces independently of each other. To show or hide the data or memory traces, use the following commands:

Type	Command
Data trace	:DISP:WIND{1-4}:TRAC{1-4}:STAT
Memory trace	:DISP:WIND{1-4}:TRAC{1-4}:ANN:MARK:POS:X

To copy the data trace to the memory trace, use the following command:

:CALC{1-4}:MATH:MEM

Configuring cross-trace math operations

You can perform math operations between the data and memory traces and have the results displayed as the data trace. To perform cross-trace math operations, use the following command:

:CALC{1-4}:MATH:FUNC

Configuring smoothing settings

To turn ON/OFF smoothing, use the following command:

:CALC{1-4}:SMO

The smoothing aperture is expressed as a percentage with respect to the sweep range. To set the smoothing aperture, use the following command:

:CALC{1-4}:SMO:APER

Selecting the data format

To select the measurement parameter data format, use the following command:

:CALC{1-4}:FORM

Configuring the display scale

Depending on the measurement parameter data format, you can configure the display scale in one of the following two ways:

Rectangular display formats:

When you use one of rectangular display formats, you can configure the display scale by setting the following four items:

Type	Command
Number of divisions	:DISP:WIND{1-4}:Y:DIV
Scale per division	:DISP:WIND{1-4}:TRAC{1-4}:Y:PDIV
Reference graticule line	:DISP:WIND{1-4}:TRAC{1-4}:Y:RPOS
Reference graticule line value	:DISP:WIND{1-4}:TRAC{1-4}:Y:RLEV

NOTE The number of divisions is a channel-wide setting (shared among all traces), while the remaining three settings are trace-specific.

You can show or hide graticule label (the label on the left-hand side of the graticule lines) by issuing the following command:

:DISP:WIND{1-4}:LAB

Smith chart/Polar formats:

When you are using one of Smith chart/Polar formats, you can only set the full scale value (the outermost circle's value) using the following command:

:DISP:WIND{1-4}:TRAC{1-4}:Y:PDIV

Auto Scale

You can use Auto Scale to automatically set the display scale. This feature works by automatically adjusting the reference division line value and the scale value per division when you are using one of the rectangular display formats or the full scale value when you are using one of Smith chart/Polar formats.

To perform Auto Scale, use the following command:

:DISP:WIND{1-4}:TRAC{1-4}:Y:AUTO

Displaying a message in the echo window

You can display a message in the echo window by issuing the following command:

:DISP:ECHO

You can clear any message displayed in the echo window by issuing the following command:

:DISP:ECHO:CLE

Turning ON/OFF display update

To turn ON/OFF the update of the LCD screen, use the following command:

:DISP:ENAB

Showing/hiding frequencies

To show or hide frequencies on the LCD screen, use the following command:

:DISP:ANN:FREQ

Showing or hiding the title

To show or hide the title, use the following command:

:DISP:WIND{1-4}:TITL

To define the title string that appears in the title display area, use the following command:

:DISP:WIND{1-4}:TITL:DATA

Configuring date/time display

To show or hide the current date and time on the right-hand side of the instrument status bar, use the following command:

:DISP:CLOC

To set the date and time, use the following command:

:SYST:DATE

:SYST:UPR

Turning ON/OFF the LCD backlight

To turn ON/OFF the LCD backlight, use the following command (note that turning OFF the backlight makes the screen unreadable):

:SYST:BACK

Setting Display Color

Selecting Display Mode

You can select one of the two LCD display modes: normal display (black background) or inverted display (white background).

To select the display mode, use the following command:

:DISP:IMAG

Setting display color for each item

To set the display colors, use the following commands:

Data trace	:DISP:COL{1-2}:TRAC{1-4}:DATA
Memory trace	:DISP:COL{1-2}:TRAC{1-4}:MEM
Graph	:DISP:COL{1-2}:GRAT{1-2}
Limit test	:DISP:COL{1-2}:LIM{1-2}
Background	:DISP:COL{1-2}:BACK

Resetting display colors to factory state

You can reset the display colors in normal display and inverted display to the preset factory state.

To reset the display colors, use the following command:

:DISP:COL{1-2}:RES

Saving and Loading the Settings

You can save the settings for measurement conditions and screen display to a file along with other instrument settings, and these settings can later be loaded from the file.

Once you have saved the measurement condition and screen display settings to a file, you can later load them whenever necessary; therefore, you can quickly modify the settings loaded from a file to create new settings without having to issue many commands.

To save the current settings to a file, use the following command:

:MMEM:STOR

To load the settings from a file, use the following command:

:MMEM:LOAD

Other topics about Setting up Analyzer

Performing Calibration

Performing Calibration

- Calibration
- Partial overwrite

Calibration

- Overview
- Performing Calibration
- Defining Calibration Kits
- Standard Definitions
- Reading/Writing Calibration Coefficient Alone
- Clearing Calibration Data and Calibration Coefficients

Other topics about Performing Calibration

Overview

You need to execute calibration to eliminate error elements related to measurement, thus allowing you to perform accurate measurement.

Performing Calibration (Obtaining calibration coefficients)

Selecting a Calibration Kit

To select a calibration kit, use the following command:

:SENS{1-4}:CORR:COLL:CKIT

Selecting a Calibration Type

The calibration coefficients are calculated based on the selected calibration type. Therefore, before you can calculate the calibration coefficients, you must select the appropriate calibration type by using one of the following commands.

Calibration type		Command
Response	OPEN	:SENS{1-4}:CORR:COLL:METH:OPEN :SENS{1-4}:CORR:COLL:METH:GPR:OPEN
	SHORT	:SENS{1-4}:CORR:COLL:METH:SHOR :SENS{1-4}:CORR:COLL:METH:GPR:SHOR
	THRU	:SENS{1-4}:CORR:COLL:METH:THRU :SENS{1-4}:CORR:COLL:METH:GPR:THRU
Enhanced Response		:SENS{1-4}:CORR:COLL:METH:ERES
1-Port		:SENS{1-4}:CORR:COLL:METH:SOLT1 :SENS{1-4}:CORR:COLL:METH:GPS1

Full 2-Port	:SENS{1-4}:CORR:COLL:METH:SOLT2

To check the currently selected calibration type, use the following command:

:SENS{1-4}:CORR:COLL:METH:TYPE?

Measuring Calibration Data

To measure the calibration data, use one of the following commands:

Calibration data items	Command
OPEN	:SENS{1-4}:CORR:COLL:OPEN :SENS{1-4}:CORR:COLL:GPAC:OPEN
SHORT	:SENS{1-4}:CORR:COLL:SHOR :SENS{1-4}:CORR:COLL:GPAC:SHOR
LOAD	:SENS{1-4}:CORR:COLL:LOAD :SENS{1-4}:CORR:COLL:GPAC:LOAD
THRU	:SENS{1-4}:CORR:COLL:THRU :SENS{1-4}:CORR:COLL:GPAC:THRU
Isolation	:SENS{1-4}:CORR:COLL:ISOL :SENS{1-4}:CORR:COLL:GPAC:ISOL

NOTE You cannot run more than one of the commands listed above at a time; if you issue another command before the currently running command completes successfully, the current command is aborted. Therefore, when you write a program that issues multiple calibration commands in series, you should use the *OPC? command or some other means to ensure that no command is executed before the preceding command completes itself.

As shown in the table below, the data required to calculate the calibration coefficients differ depending on the selected calibration type.

Calibration type (Selected ports are enclosed in parentheses)		Data				
		OPEN	SHORT	LOAD	THRU	Isolation
Response	OPEN (a)	a	Not requir	[a]	Not requir	Not require

			ed		ed	d
	SHORT (a)	Not required	a	[a]	Not required	Not required
	THRU (a-b)	Not required	Not required	Not required	a-b	[a-b]
Enhanced Response (a-b)		b	b	b	a-b	[a-b]
1-Port (a)		a	a	a	Not required	Not required
Full 2-Port (a-b)		a, b	a, b	a, b	a-b, b-a	[a-b], [b-a]

In the data section in the table, the letter m (for example, 1, a) represents the measurement data at port m; m-n (for example, 1-2, a-b) represents the measurement data between response port m and stimulus port n. You can omit data enclosed in brackets.

Calculating Calibration Coefficients

To calculate the calibration coefficients, use one of the following commands:

Calibration type	Command
Response, 1/2 port	:SENS{1-4}:CORR:COLL:SAVE

Before issuing the above commands, you must measure all required calibration data items according to your selected calibration type. Calculating the calibration coefficients clears all calibration data regardless of whether they are used for the calculation. The calibration type selection is also cleared, which results in a state where no calibration type is selected.

Turning ON/OFF Error Correction

To turn ON/OFF error correction, use the following command:

:SENS{1-4}:CORR:STAT

Also, once you have calculated the calibration coefficient using the :SENS{1-4}:CORR:COLL:SAVE command, error correction is automatically turned on.

Using ECal

An ECal (Electronic Calibration) module allows you to perform 1/2-port calibration and response (THRU) calibration without having to replace the standard device.

ECal works by using the calibration kit data contained in the ECal module instead of the calibration kit data selected for the E5061B. This means that you do not have to define or select a calibration kit when using ECal.

NOTE When two or more ECal modules are connected through the USB port, the system uses the calibration kit data of the first ECal module.

To perform ECal, use one of the following commands:

Calibration type	Command
1-Port Calibration	:SENS{1-4}:CORR:COLL:ECAL :SOLT1
Full 2-Port Calibration	:SENS{1-4}:CORR:COLL:ECAL :SOLT2
Enhanced Response Calibration	:SENS{1-4}:CORR:COLL:ECAL :ERES
Response Calibration (THRU)	:SENS{1-4}:CORR:COLL:ECAL :THRU

Simply issuing one of the above commands completes all of the tasks necessary for error correction, including measuring the calibration data, calculating the calibration coefficients, and running the error correction feature.

NOTE Once you have initiated ECal, you cannot cancel the operation.

NOTE No command entered following the initiation of ECal is processed until ECal completes successfully. Accordingly, if you issue a command that queries some data, the system does not respond to the query until ECal is complete.

The below command is intended to turn ON/OFF the isolation measurement for performing ECal. However, as the isolation performance of ENA is better than ECal, this command no longer works. ENA ignores this command.

:SENS{1-4}:CORR:COLL:ECAL:ISOL

NOTE This command takes no action and only exists to maintain backward compatibility.

To select the ECal characteristic for a user-characterized ECal, use the following command:

:SENS{1-4}:CORR:COLL:ECAL:UCH

ECal Auto-detect Function

The ECal module can automatically detect which port of the ECal module is connected to the E5061B test port. Turn OFF the auto-detect function to specify a port manually.

To turn OFF the auto-detect function, use the following command.

:SENS:CORR:COLL:ECAL:ORI

To turn OFF the auto-detect function and set a port manually, use the following command.

:SENS:CORR:COLL:ECAL:PATH

Checking the Applied Calibration Type

When you turn on the error correction, you can check the calibration type actually applied to each trace. To check the calibration type, use the following command:

:SENS{1-4}:CORR:TYPE{1-4}?

Defining Calibration kits

Selecting a Calibration Kit

To select a calibration kit, use the following command:

:SENS{1-4}:CORR:COLL:CKIT

Setting the Calibration Kit Name

To set the name of a calibration kit, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:LAB

Standard Definitions

Selecting a Standard Type

To select a standard type, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21}:TYPE

Setting the Standard Name

To set the standard name, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21}:LAB

Setting the Standard Value

To set the standard value, use one of the following commands:

Item	Command
C0	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :C0
C1	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :C1
C2	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :C2
C3	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :C3
L0	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :L0
L1	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :L1
L2	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :L2
L3	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :L3
Offset Delay	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :DEL
Offset Loss	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :LOSS
Offset Z0	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21} :Z0
Arbitrary Impedance	:SENS{1-4}:CORR:COLL:CKIT:STAN{1-21}: ARB

As shown in the table below, you need to set different items depending on the standard type.

Standard Types	C0 to C3	L0 to L3	Offset Delay	Offset Loss	Offset Z0	Arbitrary Impedance	Min. Freq.	Max. Freq.	Connector Type
OPEN	*		*	*	*		*	*	*
SHORT		*	*	*	*		*	*	*
LOAD			*	*	*		*	*	*

THRU			*	*	*		*	*	*
Arbitrary Impedance			*	*	*	*	*	*	*

You need to set the items identified by * marks in the table above.

Defining a Standard Class Assignment

To select the standard to be applied to the OPEN measurement for each port, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:ORD:OPEN

To select the standard to be applied to the OPEN measurement for Gain-Phase calibration, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:GPOR:OPEN

To select the standard to be applied to the SHORT measurement for each port, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:ORD:SHOR

To select the standard to be applied to the SHORT measurement for Gain-Phase calibration, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:GPOR:SHOR

To select the standard to be applied to the LOAD measurement for each port, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:ORD:LOAD

To select the standard to be applied to the LOAD measurement for Gain-Phase calibration, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:GPOR:LOAD

To select the standard to be applied to the THRU measurement between each pair of ports, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:ORD:THRU

To select the standard to be applied to the THRU measurement for Gain-Phase calibration, use the following command:

:SENS{1-4}:CORR:COLL:CKIT:GPOR:THRU

Saving and loading calibration coefficients

You can save calibration coefficients to a file along with other instrument settings and then later load them from the file.

By default, the system does not save calibration coefficients when it saves instrument settings. Therefore, to save calibration coefficients, you must explicitly configure the system to save them by issuing the following command:

:MMEM:STOR:STYP

To save calibration coefficients to a file, use the following command:

:MMEM:STOR

To load calibration coefficients from a file, use the following command:

:MMEM:LOAD

For more information on how to save and load calibration coefficients, refer to Saving and recalling instrument status

Reading/Writing Calibration Coefficient Alone

The calibration coefficient alone can be read from and written to the E5061B by using the following command:

:SENS{1-4}:CORR:COEF

To write a positive calibration coefficient, use one of the following commands to declare the calibration type:

:SENS{1-4}:CORR:COEF:METH:ERES

:SENS{1-4}:CORR:COEF:METH:GPR:OPEN

:SENS{1-4}:CORR:COEF:METH:GPR:SHOR

:SENS{1-4}:CORR:COEF:METH:GPR:THRU

:SENS{1-4}:CORR:COEF:METH:GPS1

:SENS{1-4}:CORR:COEF:METH:OPEN

:SENS{1-4}:CORR:COEF:METH:SHOR

:SENS{1-4}:CORR:COEF:METH:SOLT1

:SENS{1-4}:CORR:COEF:METH:SOLT2

:SENS{1-4}:CORR:COEF:METH:THRU

To validate the written calibration coefficient, use the following command:

:SENS{1-4}:CORR:COEF:SAVE

About Calibration Types and Coefficients

The following table shows the required calibration coefficients for each calibration type.

Calibration Type	Calibration Coefficient					
	ES	ER	ED	EL	ET	EX

Response calibration (OPEN)		*	*			
Response calibration (SHORT)		*	*			
Response calibration (THRU)					*	*
Enhanced response calibration	*	*	*		*	*
1-port calibration	*	*	*			
Full 2-port calibration	*	*	*	*	*	*

NOTE If either an invalid calibration coefficient is specified for the writing command or a nonexistent calibration coefficient is specified for its reading command, the following error occurs: 23, Specified error term does not exist.

Procedures for Writing Calibration Coefficient

You must follow the steps below to write the calibration coefficient.

1. Declare the calibration type to write.

Execute :SENS{1-4}:CORR:COEF:METH:xxxx command

2. Write any calibration coefficient.

Execute :SENS{1-4}:CORR:COEF command as needed for the written calibration coefficients

3. Validate the calibration coefficients.

Execute :SENS{1-4}:CORR:COEF:SAVE command

NOTE Do not execute any other command while writing the calibration coefficients. This may cause the system to function incorrectly.

Clearing Calibration Data and Calibration Coefficients

Clearing Calibration Data

You can use the following command to clear the measurement values of calibration data executed with :SENS{1-4}:CORR:COLL:OPEN command, etc.

:SENS{1-4}:CORR:COLL:CLE

These clear functions make the temporary settings during the calibration, such as trace number and measurement parameters, recover to the original state.

Clearing Calibration Coefficients

You can use the following command to clear the calibration coefficients used.

:SENS{1-4}:CORR:CLE

Partial Overwrite

- Overview
- Executing_calculation_of_calibration_coefficients_using_partial_overwrite

Other topics about Performing Calibration

Overview

The E5061B has the following calibration coefficients for full N-port calibration: Er, Es, Ed (reflection), Et (transmission), and Ex (isolation).The partial overwrite function is used to measure some of these calibration coefficients after completion of the initial calibration and then overwrite them.

The conditions under which the calibration coefficients can be calculated by the partial overwrite are as follows:

- Calibration is completed once and valid (status other than C? or C!)
- One or more measurements for re-calculation are performed.

NOTE The isolation calibration coefficient, Ex, cannot be returned to the initial value, 0, once it is calculated.

If calculation of the calibration coefficients is attempted without the measurements required to execute the partial overwrite, an error message (20: Additional Standard Needed) is displayed.

Executing calculation of calibration coefficients using partial overwrite

To calculate the calibration coefficients using partial overwrite, use the following command:

:SENS{1-4}:CORR:COLL:PART:SAVE

NOTE Before you can calculate the calibration coefficients with the partial overwrite, you must select the appropriate calibration type in the same way used for normal calibration. If calculation of the calibration coefficients is attempted without selecting the calibration type, an error message (28: Invalid Calibration Method) is displayed.

Making Measurement

Making Measurement

- Trigger System
- Starting a Measurement Cycle (triggering the instrument)
- Waiting for the End of Measurement
- Detecting Occurrence of an Error
- Improving Command Processing Speed

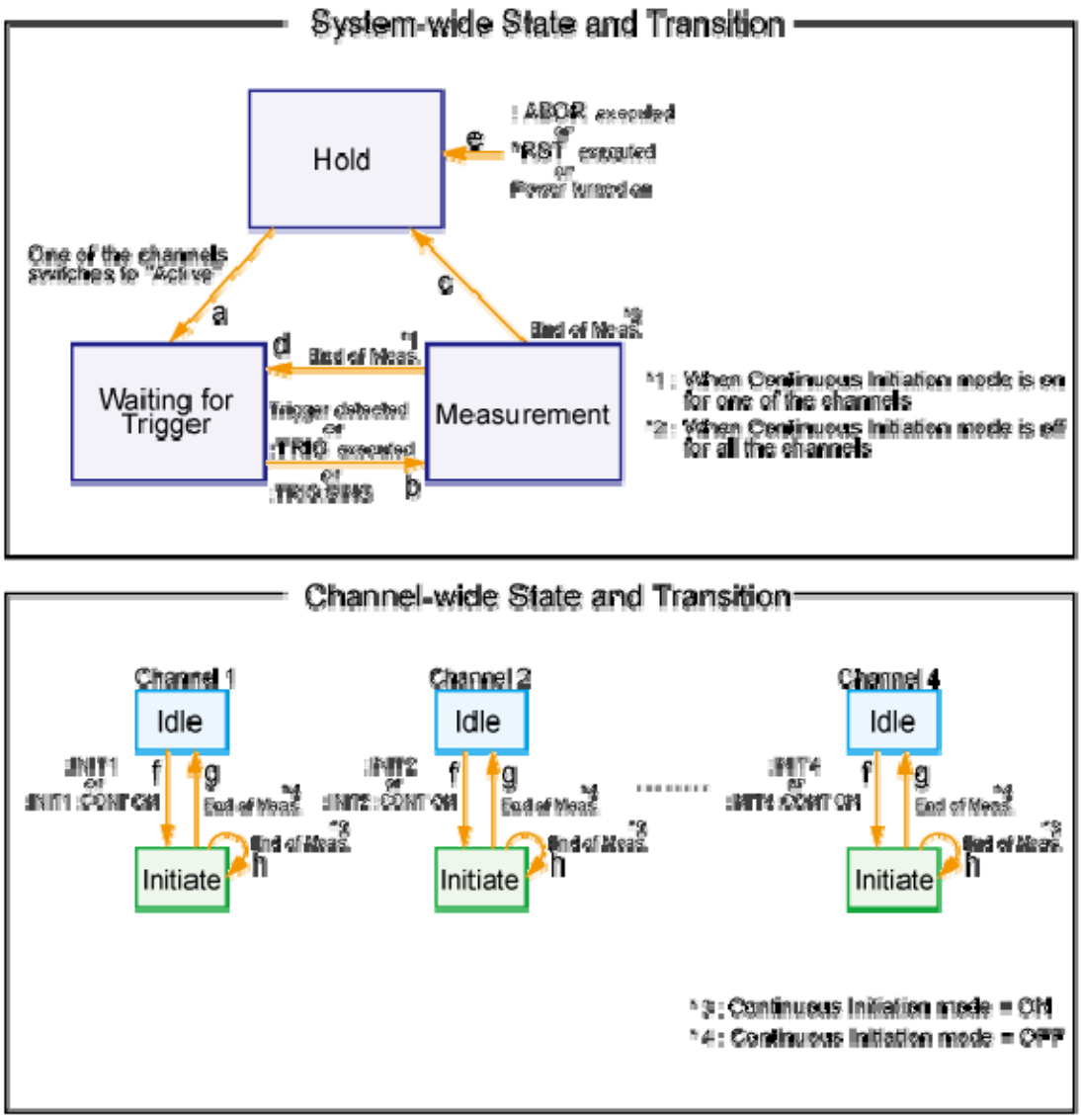
Trigger System

Other topics about Making Measurement

Overview

The trigger system is responsible for such tasks as detecting the start of a measurement cycle (triggering) and enabling/disabling measurement on each channel. As shown in the following figure, the trigger system has two types of states: system-wide and channel-wide. The system-wide state can be "Hold", "Waiting for Trigger", or "Measurement", while the channel-wide state can be "Idle" or "Initiate".

Trigger system



e5061b023

The following subsections describe each state and explains how the trigger system switches among the states.

System-Wide States and Transitions

"Hold" State

The trigger system switches to "Hold" state when one of the following commands is executed (arrow "e" in Trigger system). Also, turning ON the power to the instrument puts the trigger system into "Hold" state. When the power is turned ON, the continuous initiation mode is ON for channel 1 and the trigger source is set to "Internal". Accordingly, the trigger system immediately switches to "Waiting for Trigger" state and subsequently repeats transitions between "Measurement" and "Waiting for Trigger" states.

- :ABOR
- *RST

When the trigger system is in "Hold" state and one of the channels switches to "Initiate" state (arrow "f" in Trigger system), the trigger system switches to "Waiting for Trigger" state (arrow "a" in Trigger system).

"Waiting for Trigger" State

When the trigger system is in "Waiting for Trigger" state and either the instrument is triggered (i.e., a trigger is detected) or one of the following commands is executed, the trigger system switches to "Measurement" state (arrow "b" in Trigger system)

- :TRIG
- :TRIG:SING

As shown in the table below, the instrument is triggered differently depending on which trigger source is specified. To specify the trigger source, use the :TRIG:SOUR command.

Trigger Source	How instrument is triggered
Internal trigger	The instrument is automatically triggered within itself.
External trigger	The instrument is triggered when a trigger signal is input through the Ext Trig terminal or the handler interface
Bus trigger	The instrument is triggered when the *TRG command is issued.
Manual trigger	The instrument is triggered when you press **Trigger** > **Trigger** on the front panel.

"Measurement" State

In "Measurement" state, the instrument waits for the elapse of the sweep delay time (set by the :SENS{1-4}:SWE:DEL) and then starts a measurement cycle. This process is performed sequentially on each of those channels that are in "Initiate" state immediately before the transition to this state, in ascending order of channel number.

When the instrument has finished measuring all the active channels, the trigger system behaves in one of the following ways depending on the setting of the continuous initiation mode.

If continuous initiation mode is OFF for all channels:

The trigger system switches to "Hold" state (arrow "c" in Trigger system).

If continuous initiation mode is ON for one of the channels:

The trigger system switches to "Waiting for Trigger" state (arrow "d" in Trigger system).

Channel-wide States and Transitions

"Idle" State

A channel switches to "Initiate" state when one of the following commands is executed (arrow "f" in Trigger system).

- :INIT{1-4}
- :INIT{1-4}:CONT("ON" specified)

"Initiate" State

A channel in this state is measured just before the entire system switches to "Measurement" state.

When the instrument has finished measuring a channel, the channel behaves in one of the following ways depending on the setting of the continuous initiation mode (set by the :INIT{1-4}:CONT).

- If continuous initiation mode is OFF: The channel switches to "Idle" state (arrow "g" in Trigger system).
- If continuous initiation mode is ON: The channel remains in "Initiate" state (arrow "h" in Trigger system).

Starting a Measurement Cycle (Triggering the Instrument)

- Configuring the Instrument
- Starting Measurement on Demand

Other topics about Making Measurement

Configuring the Instrument to Automatically Perform Continuous Measurement

1. Use the :INIT{1-4}:CONT command to turn ON the continuous initiation mode for the channels you want to measure and turn the mode OFF for any other channel.
2. Issue the :TRIG:SOUR command to set the trigger source to Internal trigger.

Starting Measurement on Demand

1. Use the :INIT{1-4}:CONT command to turn ON the continuous initiation mode for the channels you want to measure and turn the mode OFF for any other channel.
2. Issue the :TRIG:SOUR command to set the trigger source to "Bus Trigger".
3. Trigger the instrument whenever you want to perform the measurement. An external controller can trigger the instrument by using one of the following three commands:

Command	Can *OPC? command be used to wait for end of sweep?	Applicable trigger source
*TRG	No	Bus trigger only
:TRIG		External trigger Bus trigger Manual trigger
:TRIG:SING	Yes	

4. Repeat step 3 to start the next measurement cycle.

Waiting for the End of Measurement

- Using the Status Register
- Using :TRIG:SING Command
- Using Wait Time

Other topics about Making Measurement

Using the Status Register

The status of the E5061B can be detected through the status registers. This section explains how to detect the end of measurement by using the status registers.

Measurement status is reported by the operation status condition register. An SRQ (service request) is useful when creating a program that uses the information reported by this register to detect the end of measurement.

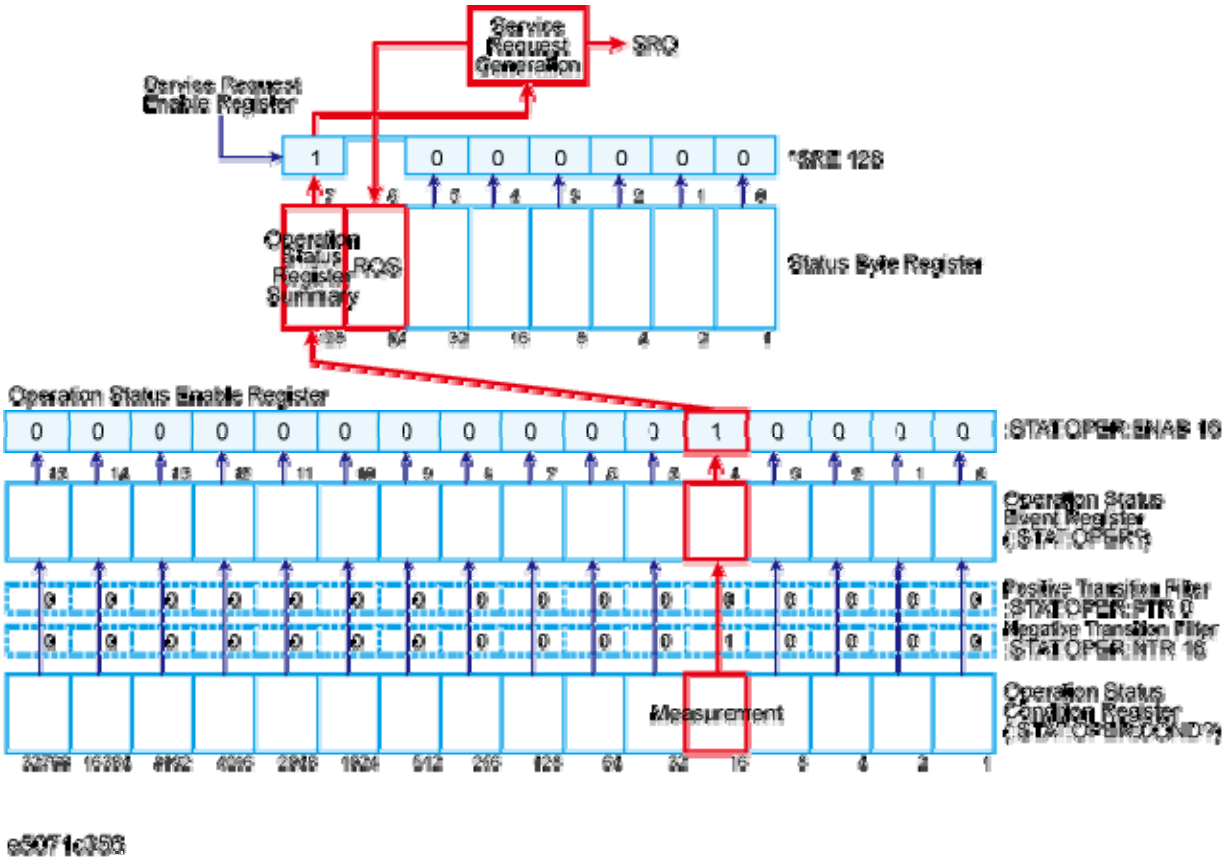
To detect the end of measurement via an SRQ, use one of the following commands:

- *SRE
- :STAT:OPER:ENAB
- :STAT:OPER:PTR
- :STAT:OPER:NTR

Follow these steps to utilize an SRQ:

1. Configure the E5061B so that it generates an SRQ when the operation status condition register's bit 4 (a bit that is set to 1 during measurement) is changed from 1 to 0.
2. Trigger the instrument to start a measurement cycle.
3. When an SRQ is generated, the program interrupts the measurement cycle.

SRQ generation sequence (at end of measurement)



Sample Program

See the Waiting for Trigger (SRQ).

Using :TRIG:SING Command

When you trigger the instrument by issuing the :TRIG:SING command, you can use the *OPC command to wait until the measurement cycle is completed.

Sample Program

See the Waiting for Trigger (OPC?).

Using Wait Time

Before creating your program, actually measure the time between the start and end of the measurement cycle. Then code your program so that the controller waits for the actual measured time by using the appropriate command (for example, the WAIT command for HTBasic). This is a straightforward method, but care must be taken: an incorrect wait time could result in an unexpected error.

Detecting Occurrence of an Error

Other topics about Working with Automatic Test System

Using Status Reporting System

The status of the E5061B can be detected through the status registers. This section describes how to detect the end of measurement by using the status registers.

The occurrence of an error presents in the standard event status register. An SRQ (service request) is useful when you create a program that uses the information reported by this register to detect the occurrence of an error.

To detect the end of sweep via an SRQ, use one of the following commands:

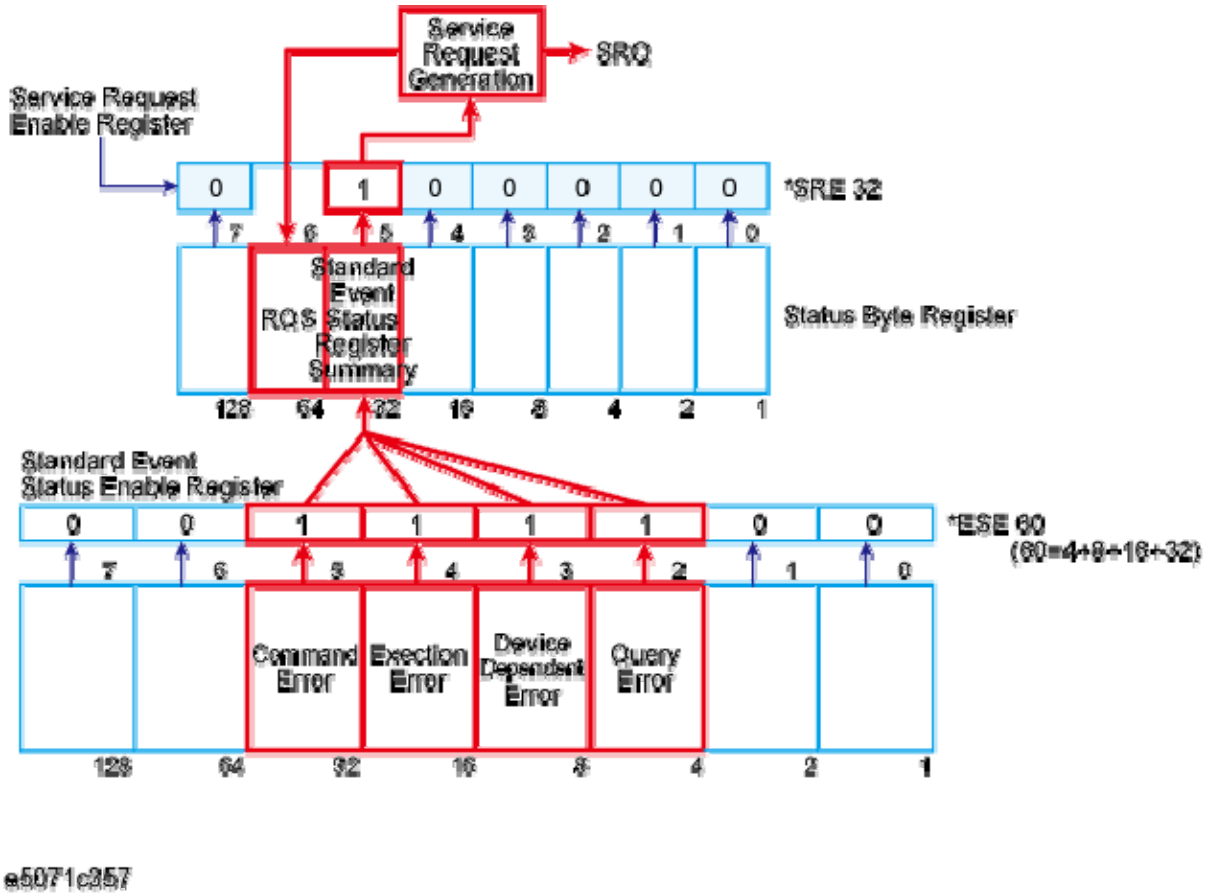
*SRE

*ESE

Follow these steps:

1. Set the E5061B so that it generates an SRQ when any of the error occurrence bits is set to 1 in the standard event status register.
2. When an SRQ is generated, the program interrupts the measurement cycle.

SRQ generation sequence (when an error occurs)



e5071c357

Using Error Queue

An error queue holds the number for the error and the error message. Reading the error queue allows the user to verify the error that has occurred. To retrieve the content of an error queue, use the following command:

:SYST:ERR?

The error queue can be used in the following ways:

1. It is used as a branch for error handling. When an error queue is retrieved, it returns 0 as the error number and "No error" as the error message if no error is detected. This can be used to detect an error and for branching the flow of a program. This is also useful when you wish to handle specific error(s). Note that this method does not allow the user to perform any processing during the occurrence of an error.
2. When an error is detected using SRQ, the error queue is used to examine the error. Refer to the sample program in this section.

Sample Program

See Error Detection (SRQ).

Improving Command Processing Speed

SCPI commands should be processed quickly to improve the throughput when such commands are frequently executed (for example, reading out traces for each measurement).

With the E5061B, the processing time for SCPI commands can be improved by decreasing the refresh rate of the LCD display.

Measurement results (trace) do not need to be updated

When the measurement trace does not need to be updated, turn OFF the updating feature of the LCD display. This improves the processing speed of SCPI commands and eliminates the time used for updating the screen.

To turn OFF the updating feature of the LCD display, use the following command:

:DISP:ENAB

Measurement results (trace) need to be updated

When the measurement trace needs to be updated, the processing speed of SCPI commands can still be improved by controlling the update timing of the LCD display:

1. Execute all SCPI commands that are required before measurement, including commands setting conditions.
2. Turn OFF the update of the LCD display.
3. Perform the measurement.
4. Execute the commands for reading out measurement result or analyzing the result. Note that reading out the result in binary format accelerates data transfer.
5. Execute the following command to update the LCD display once
   :DISP:UPD
6. Return to Step 3.

Sample program

See Control LCD Update Timing.

Reading-Writing Measurement Data

Reading-Writing Measurement Data

- Data Transfer Format
- Internal Data Processing
- Retrieving Measurement Results
- Entering Data into a Trace

Data Transfer Format

Other topics about Reading-Writing Measurement Data

Overview

When you transfer data using the one of the following commands, you can choose among ASCII transfer format, IEEE 64-bit floating point binary transfer format and IEEE 32-bit floating point binary transfer format.

NOTE The instrument always uses the ASCII transfer format when you transfer data without using any of the following commands:

- :CALC{1-4}:BLIM:REP?
- :CALC{1-4}:DATA:FDAT
- :CALC{1-4}:DATA:FMEM
- :CALC{1-4}:DATA:SDAT
- :CALC{1-4}:DATA:SMEM
- :CALC{1-4}:FUNC:DATA?
- :CALC{1-4}:LIM:DATA
- :CALC{1-4}:LIM:REP?
- :CALC{1-4}:LIM:REP:ALL?
- :CALC{1-4}:RLIM:DATA
- :CALC{1-4}:RLIM:REP?
- :SENS{1-4}:CORR:COEF
- :SENS{1-4}:CORR:COEF:GPD
- :SENS{1-4}:FREQ:DATA?
- :SENS{1-4}:SEGM:DATA

To set the data transfer format, use the following command:

:FORM:DATA

NOTE Executing the :SYST:PRES or *RST does not affect the current setting of the data transfer format.

ASCII Transfer Format

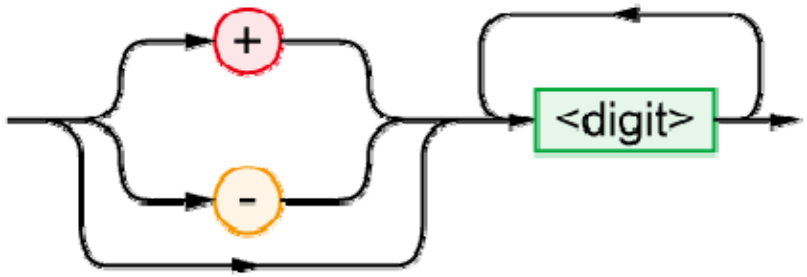
When you select the ASCII transfer format as the data transfer format, numbers are transferred as ASCII bytes, each of which corresponds to one of the formats shown below. Note that numbers are separated from one another with a comma (,) in accordance with the IEEE 488.2 specification.

NOTE Numeric data strings vary in length. Keep this in mind when you extract some data from retrieved numeric data strings in your program.

Integer Format

The figure below shows this format. Numbers are expressed as integers. For example, 201 is expressed as "+201" or "201."

Integer format

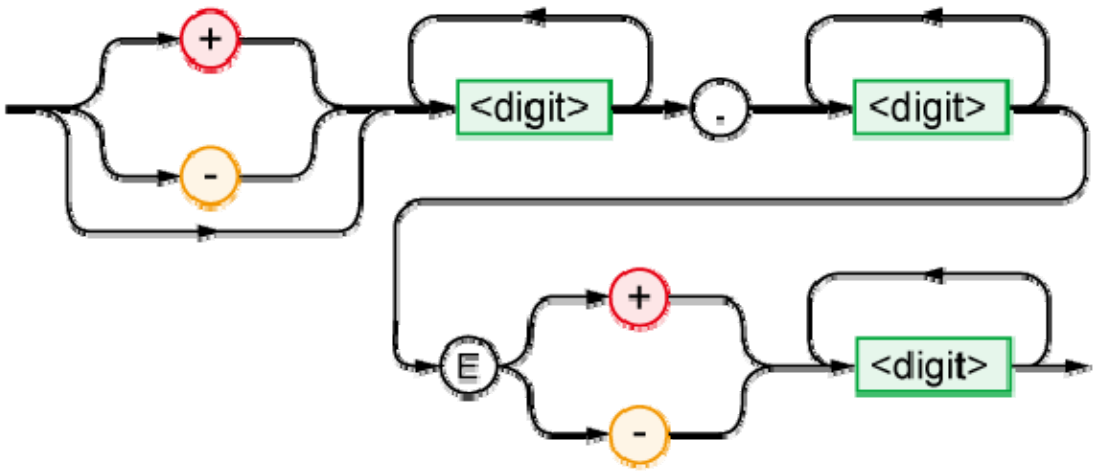


e5071c351

Floating-Point Number Format

The figure below shows this format. Numbers are expressed with floating points. The number of decimal is 12 at default. For example, 1000 is expressed as "+1.00000000000E+003." You can change the number of decimal to 14 by the SCPI.FORMat.REAL.ASCii.LENGth command.

Floating-point number format



e5071c352

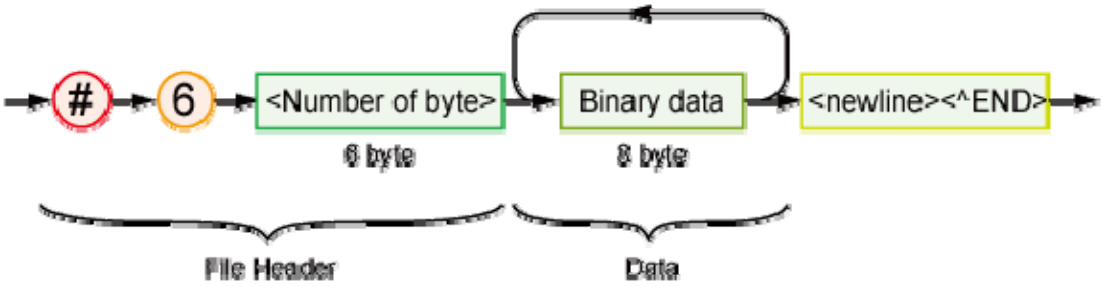
Binary Transfer Format

You can select the binary transfer format from the IEEE 64-bit floating point format or the IEEE 32-bit floating point format depending on the controller you use.

IEEE 64-bit floating point format

When you select the IEEE 64-bit floating point binary transfer format as the data transfer format, numbers are transferred in the format shown in the figure below.

Binary transfer format

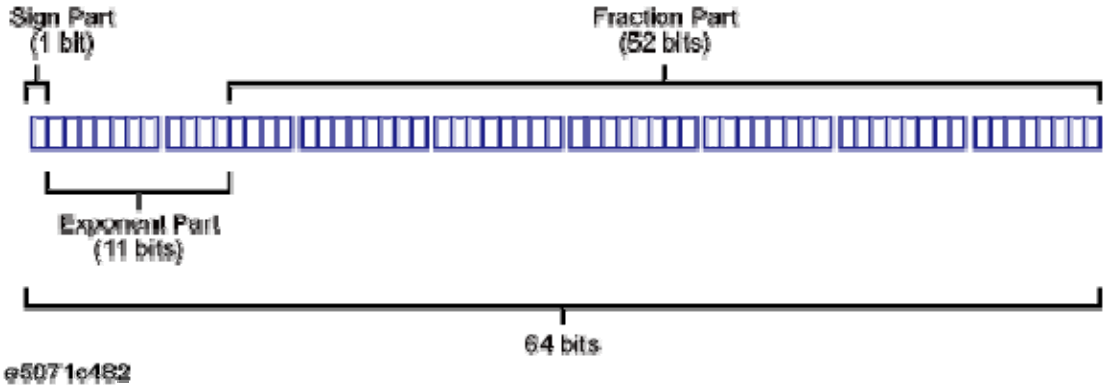


e5071c480

This data transfer format uses a header that consists of a sharp character (#), a number of 6 (which indicates the byte size of the <number of bytes transferred> part), and the <number of bytes transferred> part in this order. The header is followed by the binary data (each number consists of 8 bytes and the total is the byte size indicated by <number of bytes transferred>) and the message terminator <new line>^END.

The binary data is expressed in the IEEE 754 64-bit floating-point number format shown in the figure below.

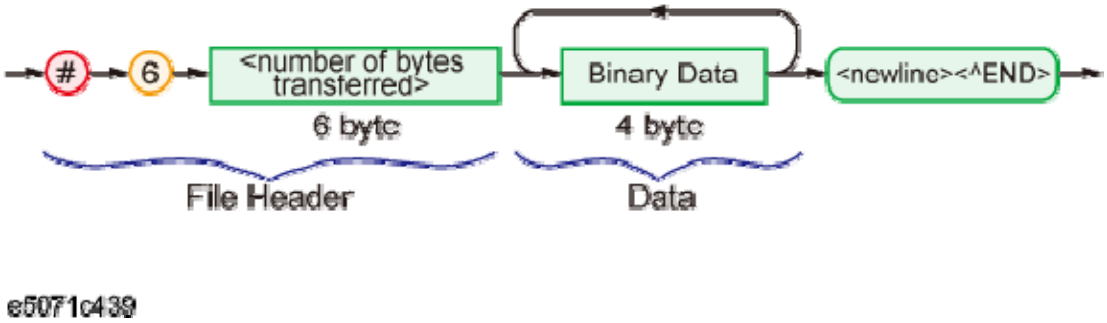
64-bit floating point format



IEEE 32-bit floating point format

When you select the IEEE 32-bit floating point binary transfer format as the data transfer format, numbers are transferred in the format shown in the figure below.

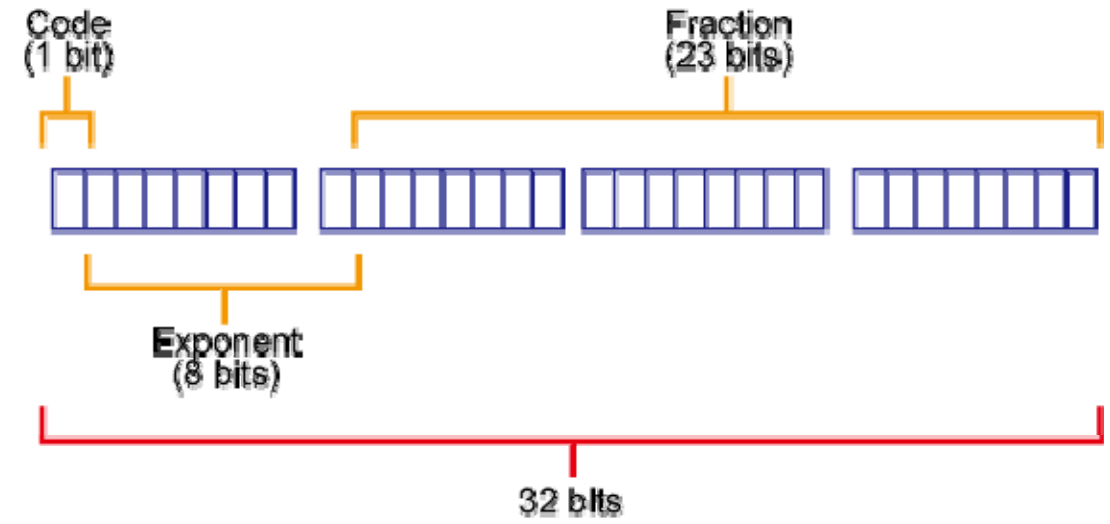
IEEE 32-bit floating point binary transfer format



This data transfer format uses a header that consists of a sharp character (#), a number of 6 (which indicates the byte size of the <number of bytes transferred> part), and the <number of bytes transferred> part in this order. The header is followed by the binary data (each number consists of 4 bytes and the total is the byte size indicated by <number of bytes transferred>) and the message terminator <new line>^END.

The binary data is expressed in the IEEE 754 32-bit floating-point number format shown in the figure below.

32-bit floating point data



e50f1c440

Byte order

When you opt to perform binary transfer, you can configure the instrument to transfer the bytes of the data in one of the following two byte orders:

NORMal

Transfer begins with the byte that contains the MSB (Most Significant Bit); that is, the leftmost byte in 64 bit floating point format and 32 bit floating point data.

SWAPped

Transfer begins with the byte that contains the LSB (Least Significant Bit); that is, the rightmost byte in 64 bit floating point format and 32 bit floating point data.

To set the byte order, use the following command:

:FORM:BORD

NOTE Executing the :SYST:PRES or *RST does not affect the current setting of the byte order.

Internal Data Processing

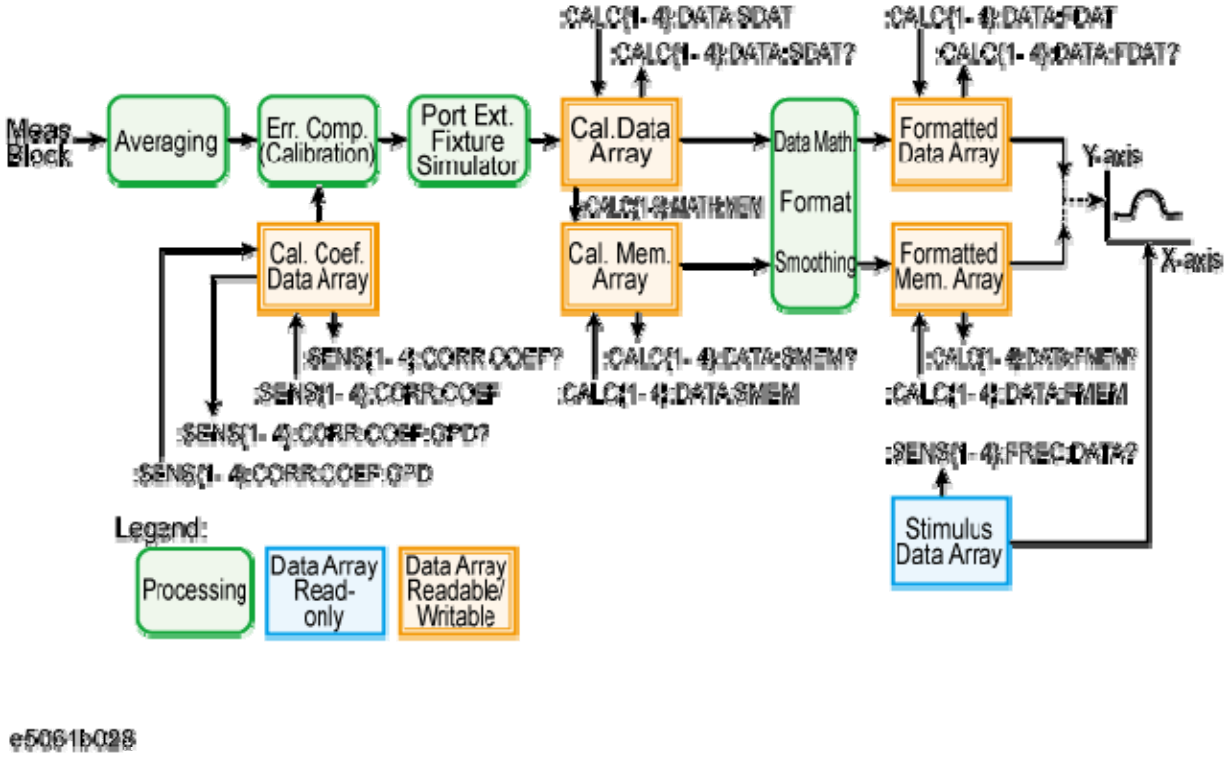
- Data Flow
- Internal Data Arrays

Other topics about Reading-Writing Measurement Data

Data Flow

The following figure provides an overview of the E5061B's internal data processing flow.

E5061B's data processing flow



e5061b028

Internal Data Arrays

Corrected data arrays

A corrected data array contains the corrected data obtained by performing error correction and port extension compensation (calibration) operations on the raw measured data of S parameter specified for each trace of each channel. Each data element is stored as a complex number (Re/Im).

The instrument retains 16 corrected data arrays at maximum, each of which is associated with one of the 4 traces contained in one of the 4 channels (4 × 4 = 16). To read/write one of the corrected data arrays, use the following command:

:CALC{1-4}:DATA:SDAT

Corrected memory arrays

When the :CALC{1-4}:MATH:MEM command is executed on a particular corrected data array, its copy is stored into the corrected memory array corresponding to that corrected data array.

The instrument retains 16 corrected memory arrays at maximum, each of which is associated with one of the 4 traces contained in one of the 4 channels (4 × 4 = 16). To read/write one of the corrected data arrays, use the following command:

:CALC{1-4}:DATA:SMEM

Formatted data array

A formatted data array contains the formatted data (values to be displayed) obtained by performing data math operations, measurement parameter conversion, and smoothing on a particular corrected data array. Regardless of the data format, it contains two data elements per measurement point as shown in the following table:

Data format	Data element (primary value)	Data element (secondary value)
log magnitude	log magnitude	Always 0
Phase	Phase	Always 0
Group delay	Group delay	Always 0
Smith chart (Lin)	Liner magnitude	Phase
Smith chart (Log)	log magnitude	Phase
Smith chart (Re/Im)	Real part of a complex number	Imaginary part of a complex number
Smith chart (R+jX)	Resistance	Reactance
Smith chart (G+jB)	Conductance	Susceptance
Polar (Lin)	Liner magnitude	Phase

Polar (Log)	log magnitude	Phase
Polar (Re/Im)	Real part of a complex number	Imaginary part of a complex number
Liner magnitude	Liner magnitude	Always 0
SWR	SWR	Always 0
Real number	Real part of a complex number	Always 0
Imaginary number	Imaginary part of a complex number	Always 0
Expanded phase	Expanded phase	Always 0
Positive phase	Positive phase	Always 0

The instrument retains 16 formatted data arrays at maximum, each of which is associated with one of the 4 traces contained in one of the 4 channels (4 × 4 = 16). To read/write one of the formatted data arrays, use the following command:

:CALC{1-4}:DATA:FDAT

Formatted memory arrays

A formatted memory array contains the formatted data (values to be displayed) obtained by performing data math operations, measurement parameter conversion, and smoothing on a particular corrected memory array.

The instrument retains 16 formatted memory arrays at maximum, each of which is associated with one of the 4 traces contained in one of the 4 channels (4 × 4 = 16). To read/write one of the formatted memory arrays, use the following command:

:CALC{1-4}:DATA:FMEM

Stimulus data arrays

A stimulus data array contains the stimulus values for all measurement points.

The instrument retains 4 stimulus data arrays at maximum, each of which is associated with one of the 4 channels. Stimulus data arrays are read-

only. To retrieve one of the stimulus data arrays, use the following command:

:SENS{1-4}:FREQ:DATA?

Calibration Coefficient Data Arrays

A calibration coefficient data array contains the calibration coefficients calculated based on the results of measurement performed with standard devices.

The instrument retains 12 calibration coefficient data arrays at maximum, each of which is associated with each channel. Commands are available for reading or writing calibration coefficient data arrays. To read or write, first use the following command:

:SENS{1-4}:CORR:COEF

NOTE If any calibration coefficient is interpolated, the interpolated calibration coefficient data array is retrieved. Once a calibration coefficient data array has been written, execute the command SENS{1-4}:CORR:COEF:SAVE to validate it.

Retrieving Measurement Results

Other topics about Reading-Writing Measurement Data

Overview

Internal data arrays allows you to retrieve all measurement results throughout a particular trace. Alternatively, markers allow you to retrieve measurement results at your specified points. For information on how to retrieve marker values, refer to Retrieving measurement results at marker positions.

Retrieving Internal Data Arrays

You can chose between the ASCII and binary data transfer formats when you retrieve internal data arrays. For more information, please refer to Data Transfer Format.

Sample Program

See Reading Data in Ascii Format and Reading Data in Binary Format.

Entering Data into a Trace

Other topics about Reading-Writing Measurement Data

Overview

You can change the data/memory trace on the LCD by writing the new data into the Formatted data array/Formatted memory arrays.

When you write data into formatted data/memory array, you can choose either the ASCII or binary transfer format (see Data Transfer Format).

Using the ASCII Transfer Format to Write Formatted Data Arrays (write_a.htb) and Using the Binary Transfer Format to Write Formatted Data Arrays (write_b.htb) show sample programs that demonstrate how to write data into formatted data arrays. The sample program in Using the ASCII Transfer Format to Write Formatted Data Arrays (write_a.htb) uses the ASCII transfer format while the sample in Using the Binary Transfer Format to Write Formatted Data Arrays (write_b.htb) uses the binary transfer format. You can find the source files of these programs, named write_a.htb and write_b.htb, on the sample program disk.

Each of the sample programs holds the sweep on channel 1, retrieves the data from a specified file (a file saved measurement data using the :MMEM:STOR:FDAT command), and populates trace 1 for channel 1 with the retrieved data.

Sample Program

See Writing Data in Ascii Format and Writing Data in Binary Format.

Analyzing Data

Analyzing Data

- Retrieving Measurement Results at Specified Points
- Searching for Positions That Match Specified Criteria
- Obtaining Limit/Bandwidth Limit/Ripple Limit Test Results
- Analysis in Time Domain (time domain function)

Retrieving Measurement Results at Specified Points

Other topics about Analyzing Data

Overview

Markers allow you to retrieve measurement results at specified points. You can use up to nine markers for each trace, and you can move them to any point on the trace. In addition to the regular markers, you can use a reference marker.

Showing/Hiding Markers

To show or hide markers, including the reference marker, use the following command:

:CALC{1-4}:MARK{1-10}

NOTE You can move markers or retrieve the data at a marker even when the markers are hidden.

NOTE The display of the reference marker is turned ON or OFF when you turn ON or OFF Reference Marker mode.

Turning ON/OFF Reference Marker Mode

Turning ON the Reference Marker mode provides relative marker values with respect to the reference marker (by subtracting the value at the reference marker from the value at a particular marker).

To turn ON or OFF Reference Marker mode, use the following command:

:CALC{1-4}:MARK:REF

Setting and Retrieving Stimulus Value at Marker Positions

To set (or change along the frequency axis) the stimulus value at a particular marker or the reference maker or to retrieve the current stimulus value, use the following command:

:CALC{1-4}:MARK{1-10}:X

When Reference Marker mode is ON, the stimulus value at a regular marker is a relative stimulus value obtained by subtracting the stimulus value at the reference marker from the actual stimulus value at that particular marker.

Retrieving Measurement Results at Marker Positions

To retrieve the measurement results (response values) at a particular marker or the reference marker, use the following command:

:CALC{1-4}:MARK{1-10}:Y?

When Reference Marker mode is ON, the response value at a regular marker is a relative value obtained by subtracting the response value at the reference marker from the actual response value at that particular marker.

Searching for Positions Matching Specified Criteria

Other topics about Analyzing Data

Overview

You can search for a position that matches specified criteria by using the Marker Search feature or analysis commands.

Using Marker Search

NOTE Marker Search is available whether the markers are shown or hidden.

Setting the Search Range

You can use either the entire sweep range or a user-defined range for the marker search range by using the following command:

:CALC{1-4}:MARK:FUNC:DOM

When you opt to use a user-defined range, use the following commands to set the range:

Description	Command
Start value (lower limit value)	:CALC{1-4}:MARK:FUNC:DOM :STAR
Stop value (upper limit value)	:CALC{1-4}:MARK:FUNC:DOM :STOP

You can also select whether to specify the marker search range independently for each trace by using the following command.

:CALC{1-4}:MARK:FUNC:DOM:COUP

Selecting a Search Type

Marker Search allows you to choose from the following eight search types:

- Maximum value
- Minimum value
- Peak (3 types)
  - Maximum peak (for a positive peak), minimum peak (for a negative peak)

- Peak nearest to the marker position on its left-hand side
- Peak nearest to the marker position on its right-hand side

▪ Target (3 types)

- Peak nearest to the marker position
- Target nearest to the marker position on its left-hand side
- Target nearest to the marker position on its right-hand side

To select a search type, use the following command:

:CALC{1-4}:MARK{1-10}:FUNC:TYPE

Defining a Peak

You can define a peak by specifying the lower limit for the peak excursion value and polarity (positive or negative peak).

To define a peak, use the following commands:

Lower limit for the peak excursion value	:CALC{1-4}:MARK{1-10}:FUNC:PEXC
Polarity	:CALC{1-4}:MARK{1-10}:FUNC:PPOL

Defining a Target

You can define a target by specifying the target value (response value) and transitional direction (positive or negative value change).

To define a target, use the following commands:

Target value	:CALC{1-4}:MARK{1-10}:FUNC:TARG
Transitional direction	:CALC{1-4}:MARK{1-10}:FUNC:TTR

Performing Marker Search

To perform Marker Search, use the following command:

:CALC{1-4}:MARK{1-10}:FUNC:EXEC

To turn ON or OFF the Search Tracking feature, which performs Marker Search every time the trace is updated, use the following command:

:CALC{1-4}:MARK{1-10}:FUNC:TRAC

Retrieving Search Results

Performing Marker Search moves the marker to the points that matches the search criteria, so you can obtain the search results by retrieving the

marker value. For information on how to retrieve marker values, refer to Setting (changing) and retrieving stimulus value at marker positions and Retrieving measurement results at marker positions.

Analysis Commands

You can use the analysis commands to perform search and analysis.

Setting the Search (Analysis) Range

You can use either the entire sweep range or a user-defined range as the search (analysis) range by using the following command:

:CALC{1-4}:FUNC:DOM

When you opt to use a user-defined range, use the following commands to set the range:

Start value (lower limit value)	:CALC{1-4}:FUNC:DOM:STAR
Stop value (upper limit value)	:CALC{1-4}:FUNC:DOM:STOP

You can also select whether to specify the marker search (analysis) range independently for each trace by using the following command:

:CALC{1-4}:FUNC:DOM:COUP

Selecting the Search (Analysis) Type

The analysis commands allow you to choose from the following five search types:

- Maximum value
- Minimum value
- Maximum peak (for a positive peak), minimum peak (for a negative peak)
- All peaks
- All targets

In addition, you can choose from the following three analysis types:

- Difference between the maximum and minimum values
- Standard deviation
- Average

To select the search (analysis) type, use the following command:

:CALC{1-4}:FUNC:TYPE

Defining a Peak

You can define a peak by specifying the lower limit for the peak excursion value and polarity (positive or negative peak).

To define a peak, use the following commands:

Lower limit for the peak excursion value	:CALC{1-4}:FUNC:PEXC
Polarity	:CALC{1-4}:FUNC:PPOL

Defining a Target

You can define a target by specifying the target value (response value) and transitional direction (positive or negative value change).

To define a target, use the following commands:

Target value	:CALC{1-4}:FUNC:TARG
Transitional direction	:CALC{1-4}:FUNC:TTR

Performing Search (Analysis)

To perform search (analysis), use the following command:

:CALC{1-4}:FUNC:EXEC

Retrieving Search (Analysis) Results

To retrieve search (analysis) results, use the following command:

:CALC{1-4}:FUNC:DATA?

The number of data items contained in search (analysis) results differ depending on the search (analysis) type and the number of points found by the search operation. To retrieve the number of data items, use the following command:

:CALC{1-4}:FUNC:POIN?

Sample Program

See Peak Search.

Notch Search

- Overview
- Setting the notch definition value
- Displaying the notch search result
- Reading out the notch search result

Other topics about Analyzing Data

Overview

The notch search function is used to obtain the bandwidth, center frequency, cutoff points (high-frequency side and low-frequency side), Q and insertion loss of a trace based on the active marker position. The notch search function starts at the left side of the active marker position, and ends when points that meet the condition are found.

- Bandwidth (BW = high – low)
- Center frequency (cent = (high +low)/2)
- Q value (Q = cent/BW)
- Loss (response value at marker position)

Where, high is the right-hand cutoff point frequency, and low is the left-hand cutoff point frequency.

NOTE For more information on notch search, see Determining the bandwidth of a trace (Notch Search)

Setting the notch definition value

The notch search function finds a point whose response value is different, by the amount defined as the notch definition value, than the response value at the marker position, and identifies that point as the cutoff point.

To set the notch definition value, use the following command:

:CALC{1-4}:MARK{1-10}:NOTC:THR

Displaying the notch search result

The following command is used to control whether to display the notch search result on the LCD:

:CALC{1-4}:MARK:NOTC

Reading out the notch search result

Once the marker is moved to an appropriate position using the marker search function or some other function, it is able to retrieve the notch search result using the following command:

:CALC{1-4}:MARK{1-10}:NOTC:DATA

NOTE It is able to retrieve the notch search result regardless of whether the marker display and the notch search result display is ON/OFF.

Analysis in Time Domain (time domain function)

- Overview
- Transforming Measurement Data to Time Domain
- Sample Program

Other topics about Analyzing Data

Overview

The time domain function provides the following functions:

- Transforming measurement data to data in the time domain (Transformation Function)
- Deleting unnecessary measurement data in the time domain (Gating Function)

Transforming Measurement Data to Time Domain

By using the Transformation Function, you can convert the results measured in the frequency domain to data in the time domain and analyze it.

ON/OFF

To turn ON or OFF the transformation function, use the following command:

:CALC{1-4}:TRAN:TIME:STAT

Selecting Transformation Type

To select the transformation type (band pass/low pass), use the following command:

:CALC{1-4}:TRAN:TIME

To select the stimulus type (impulse/step) when the transformation type is low pass, use the following command:

:CALC{1-4}:TRAN:TIME:STIM

When the transformation type is low pass, you need to execute the following command because each measurement point must be a multiple of the start frequency.

:CALC{1-4}:TRAN:TIME:LPFR

Setting Window Shape

To set the window shape, use one of the following items.

Item	Command

β	:CALC{1-4}:TRAN:TIME:KBES
Impulse width	:CALC{1-4}:TRAN:TIME:IMP :WIDT
Rise time of step signal	:CALC{1-4}:TRAN:TIME:STEP :RTIM

The above three items are dependent on each other. When the value of one of them is changed, the values of the other two are automatically changed to the corresponding values.

Unlike the manual operation, you cannot set the window shape by selecting the window type (maximum/normal/minimum). However, you can set the same shape as each window type by setting β as follows:

	Maximum	Normal	Minimum
Value of β	13	6	0

Setting Display Range

To set the display range after time domain transformation, use the following commands:

Description	Command
Start value	:CALC{1-4}:TRAN:TIME:STAR
Stop value	:CALC{1-4}:TRAN:TIME:STOP
Center value	:CALC{1-4}:TRAN:TIME:CENT
Span value	:CALC{1-4}:TRAN:TIME:SPAN

Deleting unnecessary measurement data in the time domain

You can use the Gating Function to delete unnecessary time domain data.

ON/OFF

To turn ON or OFF the gating function, use the following command:

:CALC{1-4}:FILT:TIME:STAT

Selecting Gate Type

To select the gate type, use the following command:

:CALC{1-4}:FILT:TIME

Setting Gate Shape

To select the gate shape, use the following command:
:CALC{1-4}:FILT:TIME:SHAP

Setting Gate Range

To set the gate range, use the following commands:

Description	Command
Start value	:CALC{1-4}:FILT:TIME:STAR
Stop value	:CALC{1-4}:FILT:TIME:STOP
Center value	:CALC{1-4}:FILT:TIME:CENT
Span value	:CALC{1-4}:FILT:TIME:SPAN

Sample Program

See the Time Domain.

Obtaining Limit/Bandwidth Limit/Ripple Limit Test Results

Other topics about Limit Test

Overview

You can obtain test results by issuing a result retrieval command or through the status register.

Test Results at each Measurement Point

Using commands that retrieve test results

You can obtain the test results at each measurement point by retrieving the stimulus value at failed measurement points. To retrieve failed measurement points, use the following command:

Stimulus value	:CALC{1-4}:LIM:REP?
Number of measurement points	:CALC{1-4}:LIM:REP:POIN?

Using the status register

You cannot use the status register to obtain the test results at each measurement point.

Test Results for each Trace

Using commands that retrieve test results

You can retrieve the test result for each trace (i.e., the trace-wide result that combines the results for all measurement points in a particular trace) by issuing the following command:

:CALC{1-4}:LIM:FAIL?

Using the status register

The condition register and event register under the questionable limit channel {1-4} status register provide 4 bits that correspond to traces 1 to 4 and contain the test results (0: Pass, 1: Fail) for the respective traces; for example, you can obtain the test result for trace 1 from bit 1 and that for trace 4 from bit 4.

Every bit of the condition register is set to 0 when a measurement cycle is started. Upon completion of measurement, those bits that correspond to failed traces are set to 1.

If the corresponding bit of the positive transition filter is set to 1 (preset value), each bit of the event register is set to 1 when the corresponding bit of the condition register changes from 0 to 1 (indicating that the corresponding trace failed the test).

To retrieve the registers, use the following commands:

Questionable limit channel {1-4} status register	
Condition register	:STAT:QUES:LIM:CHAN{1-4}:COND?
Event register	:STAT:QUES:LIM:CHAN{1-4}?
Questionable limit channel {1-4} extra status register	
Condition register	:STAT:QUES:LIM:CHAN{1-4}:ECH:COND?
Event register	:STAT:QUES:LIM:CHAN{1-4}:ECH?

Test Results for each Channel

Using commands that retrieve test results

No command is available that allows you to directly retrieve the test result for each channel (i.e., the channel-wide result that combines the results for all traces in a particular channel).

Using the status register

The questionable limit status event register provides 4 bits that correspond to channels 1 to 4 and contain the test results (0: Pass, 1: Fail) for the respective channels; for example, you can obtain the test result for channel 1 from bit 1 and that for channel 4 from bit 4.

Every bit of the condition register is set to 0 after the event registers are cleared by the *CLS. Upon completion of measurement, if the channel-wide test result that combines the results for all traces in a channel is fails, the corresponding bit of the condition register is set to 1.

If the corresponding bit of the positive transition filter is set to 1 (preset value), every bit of the event register is set to 1 when the corresponding bit of the condition register changes from 0 to 1.

To retrieve the registers, use the following commands:

Questionable limit status register

Condition register	:STAT:QUES:LIM:COND?
Event register	:STAT:QUES:LIM?
Questionable limit extra status register	
Condition register	:STAT:QUES:LIM:ELIM:COND?
Event register	:STAT:QUES:LIM:ELIM?

Obtaining test results for a channel (channel 1 in this example) using the status register

Overall Test Results

Using commands that retrieve test results

No command is available that allows you to directly retrieve the overall test results that combine the test results for all channels.

Using the status register

The condition register and event register under the questionable status event register provides bit 10 each, from which you can obtain the overall test result (0: Pass, 1: Fail).

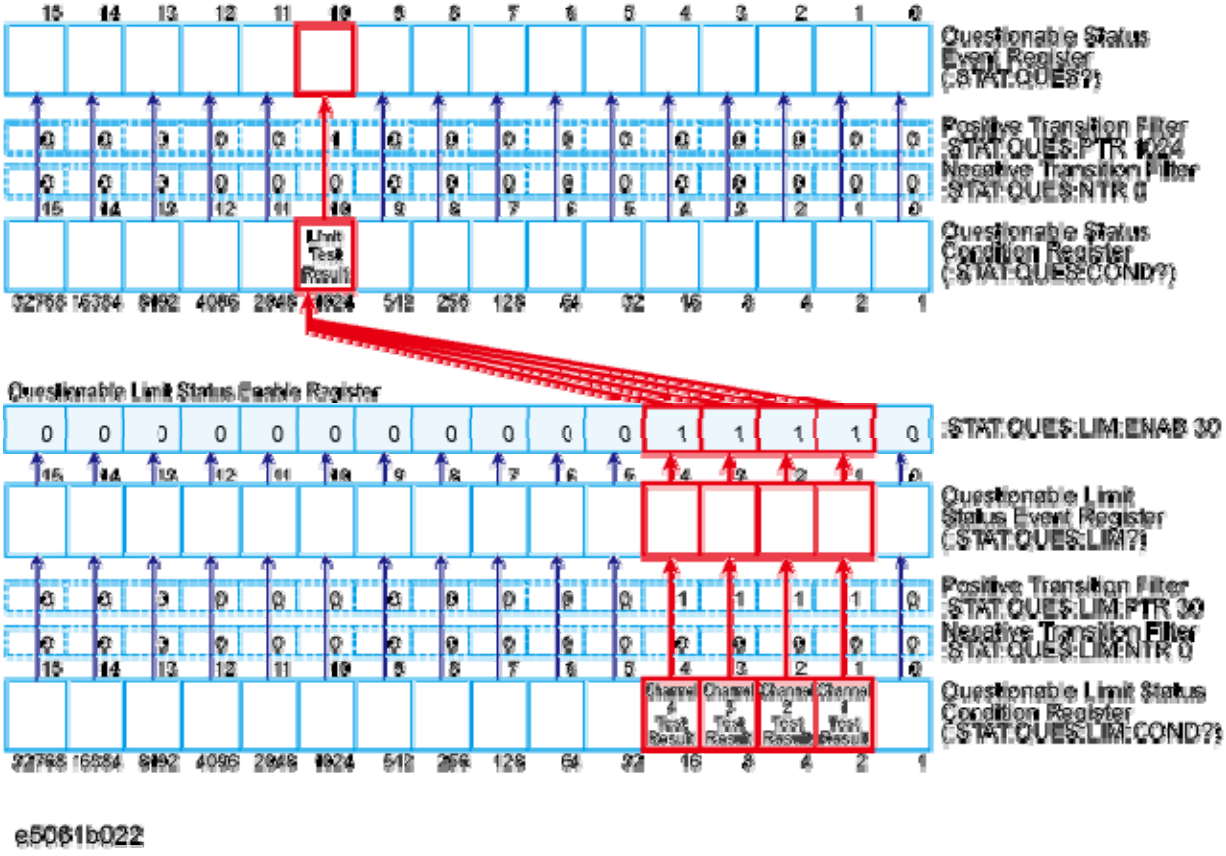
The condition register's bit 10 is set to 0 after the event registers are cleared by the *CLS. Upon completion of measurement, this bit is set to 1 if the overall test result that combines the results for all channels fails.

If the positive transition filter's bit 10 is set to 1 (preset value), the event register's bit 10 is set to 1 when the condition register's bit 10 changes from 0 to 1.

To retrieve the condition register and event register under the questionable status event register, use the following commands:

Condition register	:STAT:QUES:COND?
Event register	:STAT:QUES?

Obtaining overall test results using the status register



e5061b022

Saving and Recalling

Saving and Recalling

- Saving and Recalling File
- Managing Files

Saving and Recalling File

- Specifying File
- Saving and Recalling Instrument Status
- Saving Measurement Data
- Saving Measurement Data in Touchstone Format
- Saving Images
- Saving and Recalling Segment Sweep Table
- Saving and Recalling Limit Table
- Saving/Loading (Importing) a VBA Program

Other topics about Saving and Recalling

Specifying File

When running a command for saving, recalling, and managing files, use a filename with extension to specify a particular file. Specify "D:" at the beginning of the file name, when specifying a file at the user area of hard disk. Also, when specifying a file name with directory, use "/" (slash) or "\" (backslash) as a delimiter.

Saving and Recalling Instrument Status

You can save the instrument state using one of the following 2 methods:

- Saving the entire instrument state into a file
- Saving the state for each channel into registers A to D (volatile memory)

Selecting content to be saved

When saving the instrument status into a file or register, the content to be saved can be selected among the following 4 options:

- Instrument status only
- Instrument status and calibration coefficient array.
- Instrument status, corrected data/memory array (measurement data)
- Instrument status, calibration coefficient array, and corrected data/memory array (measurement data)

To select a content to be saved, use the following command:

:MMEM:STOR:STYP

Selecting Content to be Saved

To select whether to save the setting of all channels/traces or that of the displayed channels/traces, use the following command:

:MMEM:STOR:SALL

Saving and recalling entire instrument status

To save the entire instrument status into a file, use the following command:

:MMEM:STOR

Recalling a file saved with the above command can reproduce the status when it was saved. To recall the settings from a file, use the following command:

:MMEM:LOAD

Auto Recall

The file saved with the name autorec.sta will be automatically recalled when the E5061B is powered ON.

Saving the state for each channel into a register

For the active channel, when you want to save the instrument state specific to that channel into only one of the registers A to D, use the following command:

:MMEM:STOR:CHAN

Recalling an instrument state saved in a register can reproduce it as the state of the active channel. To recall a register, use the following command:

:MMEM:LOAD:CHAN

NOTE It is possible to recall a file from a different channel where it was saved.

The contents in the registers are lost when you turn OFF the power. You can delete (clear) the contents of all registers using the following command.

:MMEM:STOR:CHAN:CLE

Saving Measurement Data

Measurement data (in a formatted data array) can be saved to a file in CSV (Comma Separated Value) format.

To save measurement data in a file, use the following command:

:MMEM:STOR:FDAT

Executing the above command saves the measurement data of the active trace. Note that the data saved using the above command cannot be recalled from the E5061B.

Saving Measurement Data in Touchstone Format

Measurement data for the active channel can be saved to a file in touchstone format.

To determine the file type in touchstone file format and specify a port, use one of the following commands according to the number of ports used:

- :MMEM:STOR:SNP:TYPE:S1P
- :MMEM:STOR:SNP:TYPE:S2P

To set the data type of the files saved in touchstone format, use the following command:

:MMEM:STOR:SNP:FORM

To save measurement data in touchstone format, use the following command:

:MMEM:STOR:SNP

NOTE Only trace data of frequency sweep can be saved in touchstone format files. Trace data of power sweep measurement cannot be saved.

Saving Images

Images displayed on the LCD screen can be saved to a file in the bitmap (.bmp) or portable network graphics (.png) format.

To save the screen image to a file, use the following command:

:MMEM:STOR:IMAG

Executing the above command saves the screen image when the command is invoked.

NOTE This gives different screen image results from those obtained by pressing the **Capture** key on the front panel.

Saving and Recalling Segment Sweep Table

Segment sweep table can be saved in the file with CSV (Comma Separated Value) format.

To save segment sweep table on a file, use the following command:

:MMEM:STOR:SEGM

Executing the above command saves the segment sweep table for the active channel.

Recalling the file saved using the above command can reproduce the segment sweep table on the active channel.

To recall the settings from a file, use the following command:

:MMEM:LOAD:SEGM

NOTE It is possible to recall a file from a different channel where it was saved. Note that recalling operation is not guaranteed for the file that might have been modified with editor.

Saving and Recalling Limit Table

Limit table can be saved in the file with CSV (Comma Separated Value) format. To save limit table on a file, use the following command:

:MMEM:STOR:LIM

Executing the above command saves the limit table for the active trace of the active channel.

Recalling the file saved using the above command can reproduce the limit table on the active trace of the active channel. To recall the settings from a file, use the following command:

:MMEM:LOAD:LIM

NOTE It is possible to recall a file from a different channel or trace where it was saved. Note that recalling operation is not guaranteed for the file that might have been modified with editor.

Saving/Loading (Importing) a VBA Program

Saving

Only the VBA project file can be saved using command.

To save the VBA project that is opened on the VBA editor on the file, use the following command.

:MMEM:STOR:PROG

Loading (importing)

To load the VBA project to the VBA editor, or to import the module/form file, use the following command.

:MMEM:LOAD:PROG

Executing the above command loads/imports the file according to its extension as follows:

Extension	File type
vba	VBA Project
bas	Standard module

frm	User Forms
cls	Class Modules

Sample program

See the Saving Files.

Managing Files

Other topics about Saving and Recalling

Various Commands

Creating directory (folder)

To create a directory (folder), use the following command:

:MMEM:MDIR

Deleting file (directory)

To delete a file or a directory, use the following command:

:MMEM:DEL

Copying file

To copy a file, use the following command:

:MMEM:COPY

Transferring files

File transfer from the external controller to the E5061B is possible by reading data from a file on the controller and then writing them to the file on the E5061B.

:MMEM:TRAN

Also, file transfer from the E5061B to the external controller can be possible by reading data from a file on the E5061B using the commands as query and then writing them to the file on the controller.

Retrieving data from storage

To retrieve information for the storage that is built in the E5061B (usage, property of file located in a specified directory), use the following command;

:MMEM:CAT?

Sample Program

See the Transferring Files.

Communication with External Devices using I/O Port

Communication with External Devices (24 Bit I/O Port)

- 24 Bit (Handler) I/O Port Overview
- I/O Signal Pin Layout and Description
- Inputting/Outputting Data
- Preset states at power-on
- Timing Chart
- Electrical Characteristics

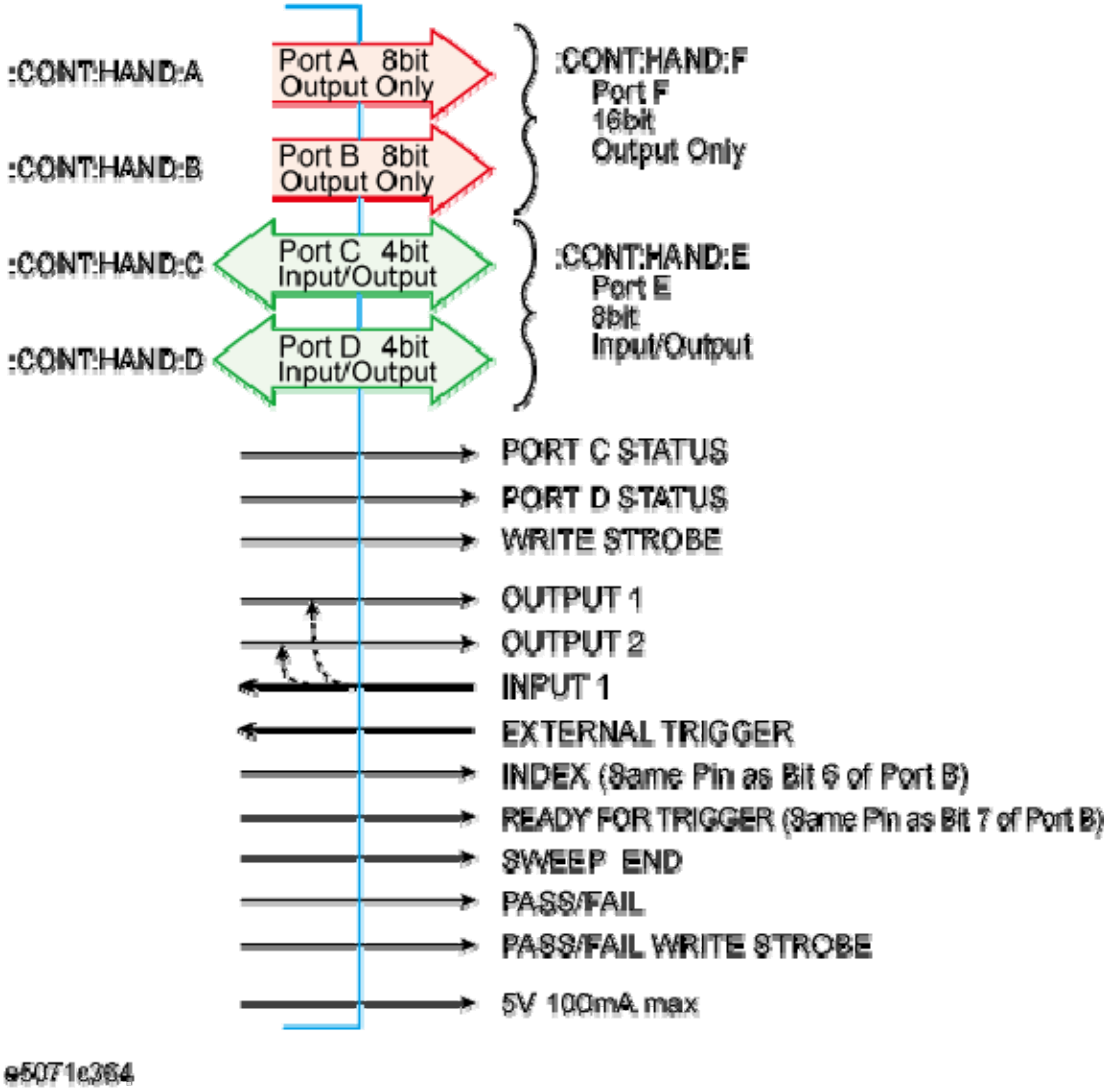
24 Bit (Handler) I/O Port Overview

The E5061B 24 Bit (handler) I/O port provides four independent parallel ports for data I/O associated with several control signal lines and the power line. All signals operate in TTL logic.

The data I/O ports are configured with 2 pairs of 8 bit output port and 2 pairs of 4 bit bi-directional port. Also those ports can cooperate to provide a maximum 16-bit-width output port or a maximum 8-bit-width input port.

The I/O signals operate on the negative logic basis, which can be altered. The control signal lines consist of various control output data, including completion of measurement or control signal for handshaking.

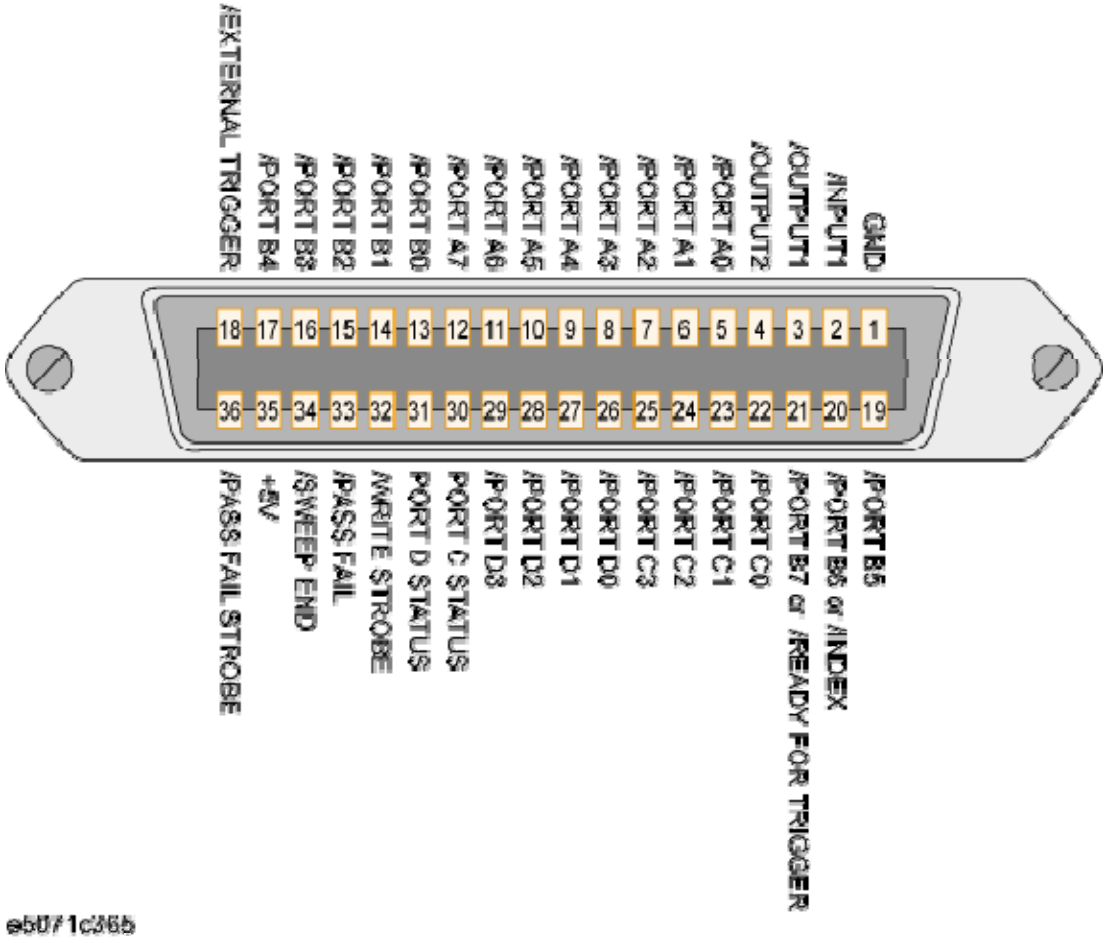
I/O ports and control signal lines



Other topics about Communication with External Devices

I/O Signal Pin Layout and Description

The layout of the I/O signal pins on the handler interface connector and its description are shown below.



A slash (/) symbol preceding signal names means that they are negative logic (active low).

Pin number	Signal name	Input/Output	Description
1	GND	N/A	Ground.
2	/INPUT1	Input	When this port receives a negative pulse, /OUTPUT1 and /OUTPUT2 are changed to the Low level.
3	/OUTPUT1	Output	Changes to the Low level when /INPUT1 receives a

			negative pulse. A command is available for altering the Low/High level logic.
4	/OUTPUT2	Output	Changes to the Low level when /INPUT1 receives a negative pulse. A command is available for altering the Low/High level logic.
5	/PORT A0	Output	Bit 0 of port A (8 bit parallel output port)
6	/PORT A1	Output	Bit 1 of port A.
7	/PORT A2	Output	Bit 2 of port A.
8	/PORT A3	Output	Bit 3 of port A.
9	/PORT A4	Output	Bit 4 of port A.
10	/PORT A5	Output	Bit 5 of port A.
11	/PORT A6	Output	Bit 6 of port A.
12	/PORT A7	Output	Bit 7 of port A.
13	/PORT B0	Output	Bit 0 of port B (8 bit parallel output port)
14	/PORT B1	Output	Bit 1 of port B.
15	/PORT B2	Output	Bit 2 of port B.
16	/PORT B3	Output	Bit 3 of port B.
17	/PORT B4	Output	Bit 4 of port B.
18	/EXTERNAL TRIGGER	Input	An external trigger signal. When the trigger source is set to the "External," this port generates a trigger in respond to the trailing edge of a negative pulse.

19	/PORT B5	Output	Bit 5 of port B.
20	/PORT B6	Output	Bit 6 of port B.
	/INDEX		Indicates that analog measurement is complete. The /INDEX signal changes to the Low level when analog measurement (all sweeps of all channels) is complete. When the handler receives the signal, it assumes that it is ready to connect the next DUT. However, no measurement data is available until data calculation is completed. When the point trigger function is ON, it goes to the High level before starting measurement of the first measurement point and returns to the Low level after completing measurement of all measurement points.
21	/PORT B7	Output	Bit 7 of port B.
	/READY FOR TRIGGER		Indicates that the instrument is ready for triggering. This signal is changed to the Low level when the instrument is ready to receive a trigger signal. The /READY FOR TRIGGER signal goes to the Low level when the instrument is ready to accept the trigger signal for the first

			point and goes to the High level when the trigger signal for the first point is received. **When the point trigger is OFF:** When measurement of all measurement points is completed and the instrument is ready to receive the trigger signal for the first point of the next sweep, this signal goes to the Low level again. **When the point trigger is ON:** When each measurement point is completed and the instrument is ready to receive the trigger signal for the next measurement point, this signal goes to the Low level again.
22	/PORT C0	Input/Output	Bit 0 of port C (4 bit parallel I/O port)
23	/PORT C1	Input/Output	Bit 1 of port C.
24	/PORT C2	Input/Output	Bit 2 of port C.
25	/PORT C3	Input/Output	Bit 3 of port C.
26	/PORT D0	Input/Output	Bit 0 of port D (4 bit parallel I/O port)
27	/PORT D1	Input/Output	Bit 1 of port D.
28	/PORT D2	Input/Output	Bit 2 of port D.
29	/PORT D3	Input/Output	Bit 3 of port D.
30	PORT C STATUS	Output	Port C status signal. This signal is changed to the High level when the port C

			is configured to output port. It is changed to the Low level when the port is configured to input port.
31	PORT D STATUS	Output	Port D status signal. This signal is changed to the High level when the port D is configured to output port. It is changed to the Low level when the port is configured to input port.
32	/WRITE STROBE	Output	A output port write strobe signal. When data is present (that is, output level changes) on any of the output ports, this signal provides a negative pulse.
33	/PASS FAIL	Output	Each limit test's results signal. This signal changes to the High level when limit test, bandwidth test, or ripple test results return FAIL. It changes to the Low level when all limit test results return PASS.
34	/SWEEP END	Output	A sweep completion signal. When measurement (all sweeps of all channels) and data calculation are completed, this signal provides a negative pulse.
35	+5V	Output	Provides +5V DC power supply for external instruments.
36	/PASS FAIL	Output	Each limit test's results write a strobe signal.

	STROBE		When limit test result is present on /PASS FAIL, this signal provides a negative pulse.

Other topics about Communication with External Devices

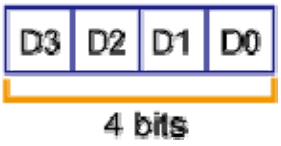
Inputting/Outputting Data

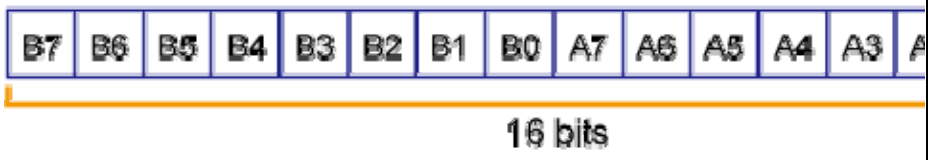
- Overview
- Specifying Signal Direction of Port
- Reading Data Input from Port
- Data Output to Port
- Sample Program

Other topics about Communication with External Devices

Overview

The E5061B 24 Bit (Handler) I/O port provides the ports for data I/O shown below.

Port Name	Usage	Data Structure
Port A	Output	
Port B	Output	
Port C	Input/Output	
Port D	Input/Output	

Port E	Input/Output	
Port F	Output	

Specifying Signal Direction of Port

Signal direction (input/output) can be changed for the ports C, D, and E as shown in I/O ports and control signal lines. Thus, before the ports are used, the directions should be determined according to their usage.

To specify the signal direction for the ports C and D, use the following command. Direction for the port E depends on the setting for the ports C and D.

Port Name	Command
Port C	:CONT:HAND:C:MODE
Port D	:CONT:HAND:D:MODE

Reading Data Input into Port

When the ports C, D, or E are configured to input ports, binary data represented by High(0)/Low(1) of each bit of the port is read as decimal data.

To retrieve the data, use the following command as query:

Port Name	Command
Port C	:CONT:HAND:C
Port D	:CONT:HAND:D
Port E	:CONT:HAND:E

Data Output to Port

To ports A through F (the ports C, D, and E should be configured to output ports), binary data (decimal data when output data is specified with a

command) represented by High(0)/Low(1) of each bit of the port can be output.

To output data, use the following command:

Port Name	Command
Port A	:CONT:HAND:A
Port B	:CONT:HAND:B
Port C	:CONT:HAND:C
Port D	:CONT:HAND:D
Port E	:CONT:HAND:E
Port F	:CONT:HAND:F

NOTE The bit 6 of the data output by :CONT:HAND:B (the bit 14 of the data output by :CONT:HAND:F) is ignored when outputting the /INDEX signal is turned ON.

NOTE The bit 7 of the data output by :CONT:HAND:B (the bit 15 of the data output by :CONT:HAND:F command) is ignored when outputting the /READY FOR TRIGGER signal is turned ON.

Sample Program

See Handler Interface.

Preset states at power-on

The 24 bit (Handler) I/O port is set at power-on as follows (not affected at reset)

Description	Status
Port A	High (All Bits)
Port B	High (All Bits)
Port C	Input
Port D	Input
Port C STATUS	Low
Port D STATUS	Low
/OUTPUT1	High
/OUTPUT2	High
/SWEEP END	High
/PASS FAIL	High

Other topics about Communication with External Devices

Timing Chart

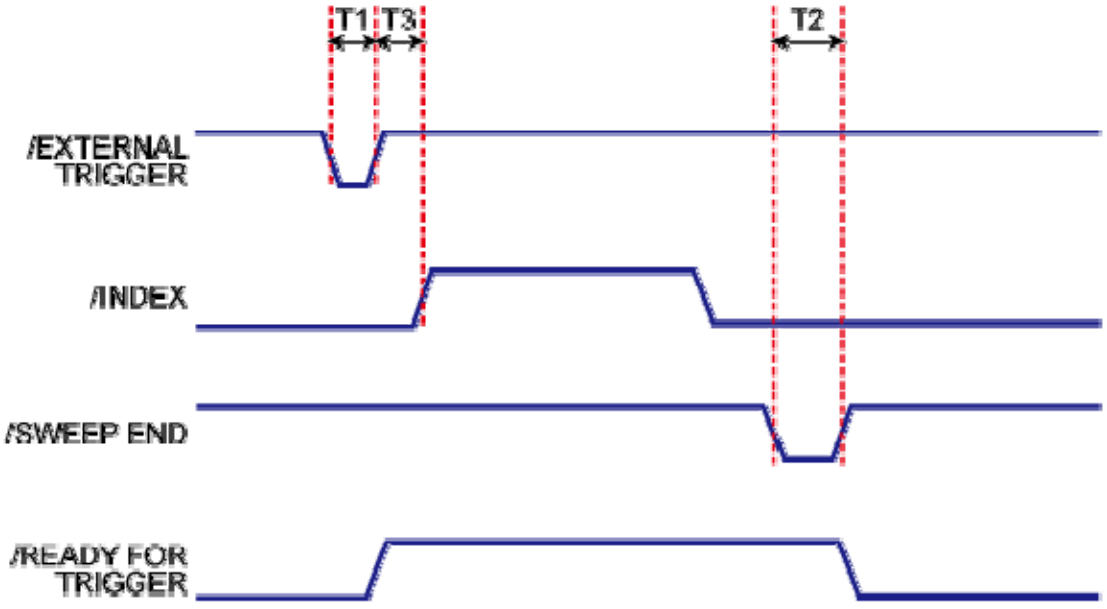
- Overview
- Timing Chart of I/O Port Signal
- Timing Chart of Data Output and Write Strobe Signal
- Timing Chart of Limit Test Result Output and Write Strobe Signal
- Timing Chart of /INPUT1 and /OUTPUT1, /OUTPUT2

Other topics about Communication with External Devices

Overview

This section shows the typical timing chart of I/O port Signal.

Timing chart of I/O Port Signal (Point Trigger: off)



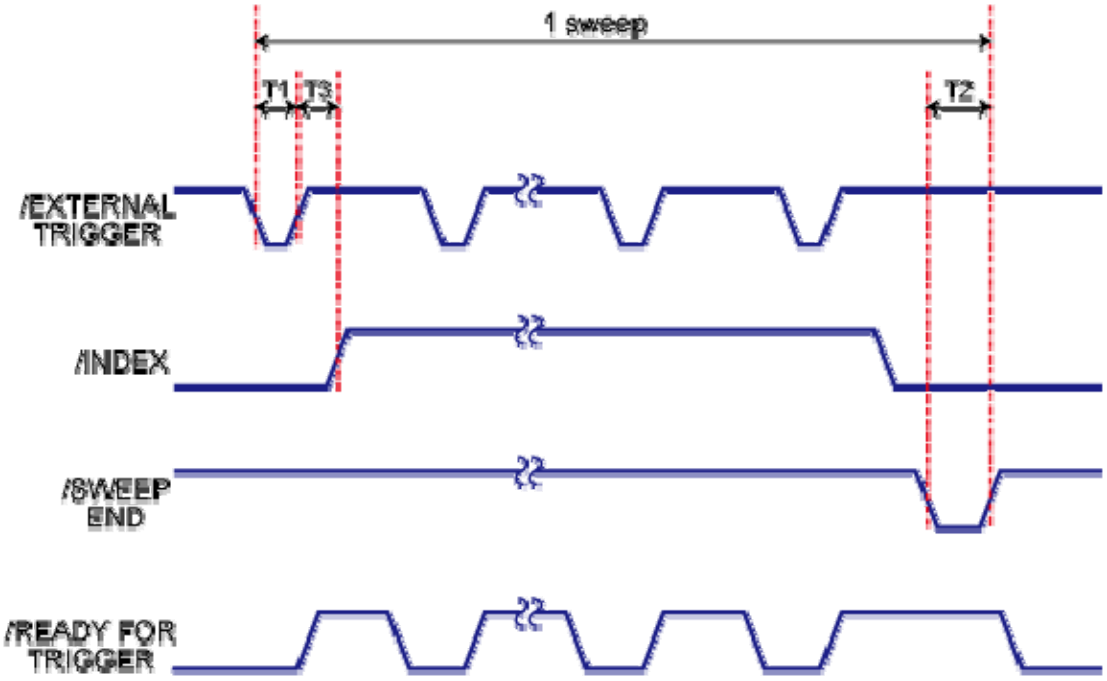
e5061b040

		Minimum value	Typical Value	Maximum value
T1	Pulse width of /EXTERNAL TRIGGER or External Trigger Input Port	1 µs	-	-
T2	Pulse width of /SWEEP END	10 µs	12 µs	-

T3	Time set as the trigger delay time	-	(See below note)	-

NOTE The Trigger Delay Time (T3) is not constant, because it is time that the user sets.

Timing Chart of I/O Port Signal (Point Trigger: on)



e5081b041

When the point trigger function is ON, the /EXTERNAL TRIGGER signal must be inputted for each measurement point during a single sweep. The /INDEX signal goes to the High level before starting measurement of the first measurement point and returns to the Low level after the completing measurement of all measurement points.

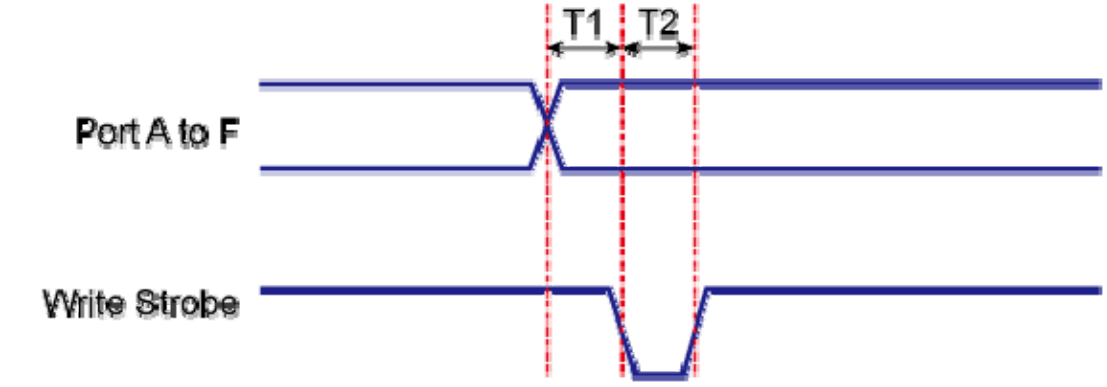
The /READY FOR TRIGGER signal goes to the Low level when the instrument is ready to accept the trigger signal for the first point and then goes to the High level when the trigger signal for the first point is received.

When measurement of all measurement points is completed and the instrument is ready to receive the trigger signal for the first point of the next sweep, this signal goes to the Low level again.

When the point trigger function is ON, the /READY FOR TRIGGER signal goes High each time a trigger signal is received and goes Low when measurement of each measurement point is completed and the instrument is ready to accept a trigger for the next measurement point.

The times of T1 and T2 are the same as those when the point trigger function is OFF. For more information, see Timing chart of I/O Port Signal (Point trigger function:OFF).

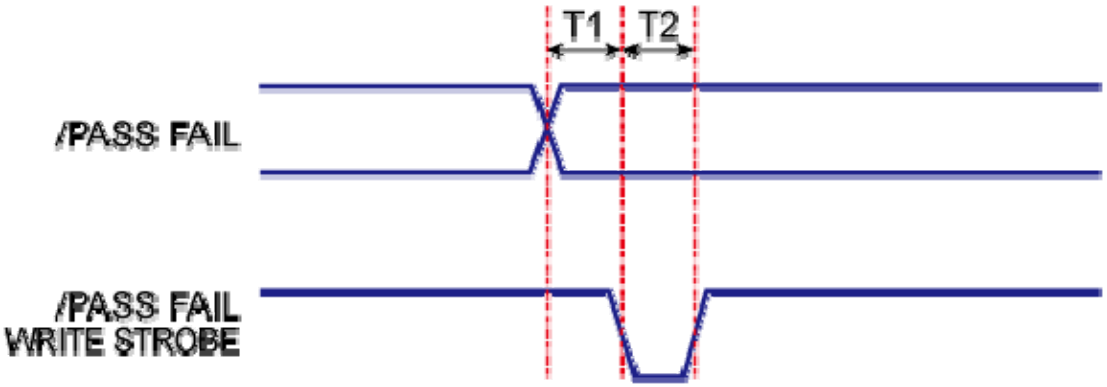
Timing Chart of Data Output and Write Strobe Signal



e5071c366

T1	Response time of write strobe signal	1 µs
T2	Pulse width of write strobe signal	1 µs

Timing Chart of Limit Test Result Output and Write Strobe Signal

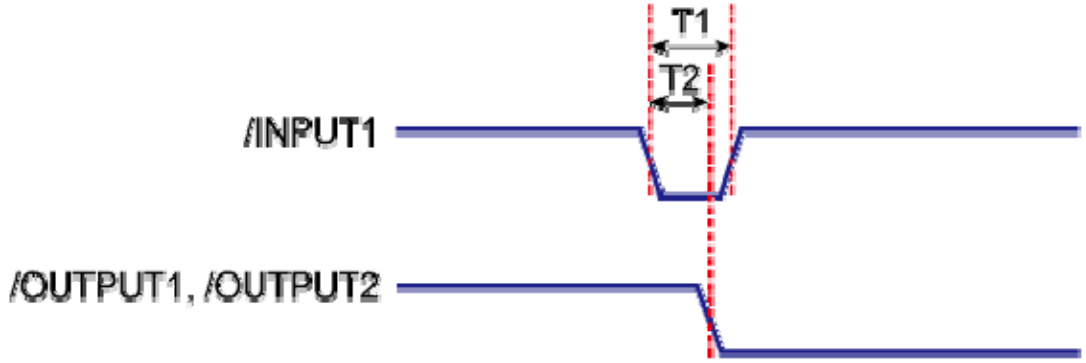


e5071c367

T1	Response time of /PASS FAIL write strobe	1 µs
T2	Pulse width of /PASS FAIL write strobe	1 µs

NOTE When the average trigger function is activated, the fail and write strobe signals are output at the time that the average test result shows "failed" on a certain channel.

Timing Chart of /INPUT1 and /OUTPUT1, /OUTPUT2



e50f1e369

		Minimum value	Maximum value
T1	Pulse width of /INPUT1	1 µs	-
T2	Response time of /OUTPUT1, /OUTPUT2	0.2 µs	0.4 µs

Electrical Characteristics

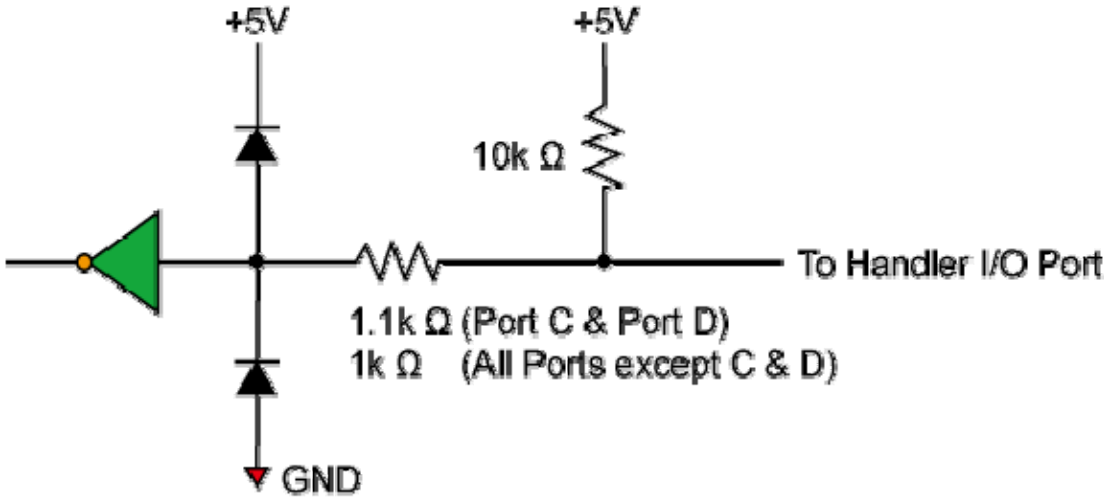
- Input Signal
- Output Signal
- Power Supply (+5 V)

Other topics about Communication with External Devices

Input Signal

All input signals are TTL compatible.

Maximum rate input voltage		-0.5 V to 5.5 V
Input voltage	**High level**	2.0 V to 5.0 V
	Low level	0 V to 0.5 V



e5071c370

Output Signal

All output signals are TTL compatible.

Maximum rate output current		-10 mA to 10 mA
Output current	**High level**	-5 mA

	Low level	3 mA
Output voltage	**High level**	2.0 V to 3.3 V (when output current is from -5 mA to 0 mA) 3.20 V (when output current is -1 mA) 2.75 V (when output current is -5 mA)
	Low level	0 V to 0.8 V (when output current is from 0 mA to 3 mA) 0.25 V (when output current is 1 mA) 0.55 V (when output current is 3 mA)

Power Supply (+5 V)

The following table shows electrical characteristics of +5 V power supply for external instruments.

Output voltage	4.5 V to 5.5 V
Maximum output current	100 mA

Status Reporting System

Status Reporting System

- General Status Register Model
- Using the Status Reporting System
- Status Register Structure

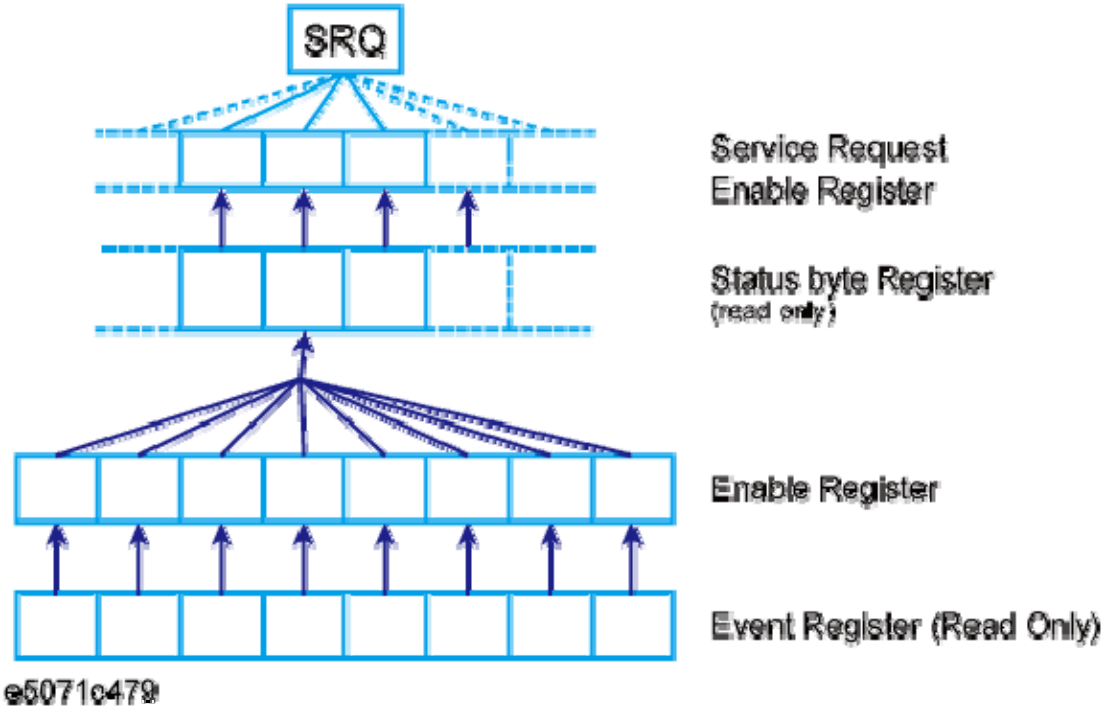
General Status Register Model

Other topics about Status Reporting System

Overview

The E5061B has a status reporting system to report the condition of the instrument.

General status register model



The status reporting system has a hierarchical structure as shown in the figure above. When the instrument satisfies a particular condition, the corresponding bit of the event register is set to 1. Therefore, you can check the instrument status by reading the event register.

When the event register bit is set to "1" and a corresponding enable register bit (a bit marked with an arrow in General status register model) is also "1," the summary bit of the status byte register is set to "1." You can read the status byte register by using the serial poll.

If the bit of the service request enable register is "1," a service request (SRQ) is generated by the positive transition of the corresponding status byte register bit. By generating SRQ, you can notify the controller that the E5061B is requesting service. In other words, interruption by SRQ can be programmed. For more information on using SRQ, see Using the status register or Using the status reporting system.

Event Register

Reflects the corresponding condition of the E5061B (e.g., occurrence of an event) as a bit status. These bits continuously monitor changes in the E5061B's state and change the bit status when the condition (e.g., change bit status to "1" if a specific event occurs) for each bit is met. You cannot change the bit status by issuing a SCPI command.

Enable Register

Setting the enable register allows you to specify event register bits that can set "1" to the summary bit of the status byte register when an event occurs. The register bits work as mask bits; setting "1" to an enable register will enable a corresponding bit in the event register.

For example, when you want to set "1" as the summary bit in the status byte register by a specific register condition, set the corresponding enable register to "1."

Status Byte Register

If the enabled event register is set to "1," a corresponding bit of the status byte register is also set to "1." This register also indicates the output queue and SRQ status.

The value of the status byte register can be read by using the *STB? command or serial poll (SPOLL statement in HTBasic) from the controller.

Reading the status byte register by using the *STB? command does not affect the contents of the status byte register. However, reading it with the SPOLL statement of HTBasic clears the RQS bit in the status byte register.

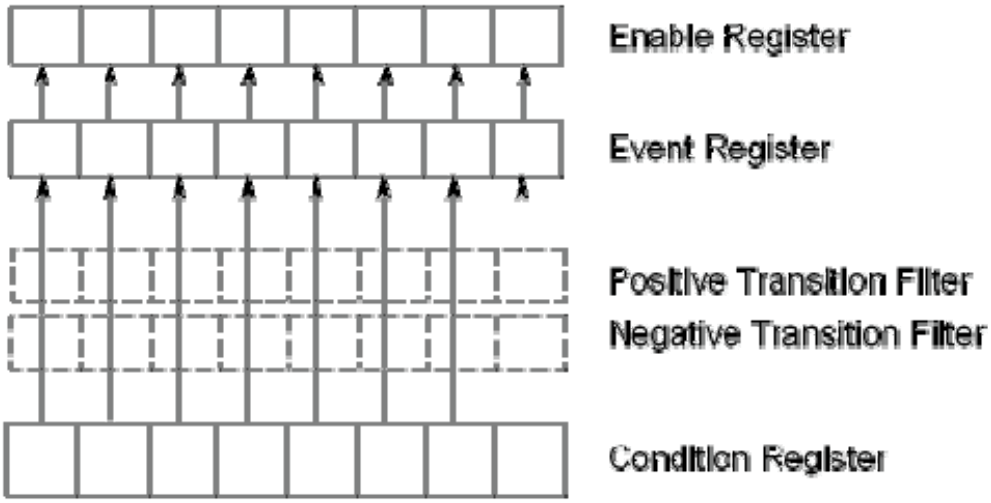
Also, setting the service request enable register using the *SRE command can generate a service request synchronously with the status byte register.

Condition Register and Transition Filter

When the status register has a transition filter, there is a lower register called a condition register under the event register. The transition filter is between the event register and the condition register.

The transition filter enables you to select a positive and/or negative transition of the condition register bit in order to set a bit in the corresponding event register. For example, using the negative transition filter to set bit 3 to "1" causes bit 3 of the event register to be set to "1"; when bit 3 of the condition register makes a negative transition, it changes from 1 to 0.

Transition filter and condition register



8294spe022

In the E5061B, the following registers provide a condition register and transition filter:

- Operation status register
- Questionable status register
- Questionable limit status register
- Questionable limit channel status register
- Questionable bandwidth limit status register
- Questionable bandwidth limit channel {1-4} status register
- Questionable ripple limit status register
- Questionable ripple limit channel {1-4} status register

Using the Status Reporting System

You can manage the status report system using the following commands in any combination:

- *CLS
- *SRE
- *STB?
- *ESE
- *ESR?
- :STAT:PRES
- :STAT:OPER:ENAB
- :STAT:OPER:COND?
- :STAT:OPER?
- :STAT:OPER:PTR
- :STAT:OPER:NTR
- :STAT:QUES:ENAB
- :STAT:QUES:COND?
- :STAT:QUES?
- :STAT:QUES:PTR
- :STAT:QUES:NTR
- :STAT:QUES:LIM:ENAB
- :STAT:QUES:LIM:COND?
- :STAT:QUES:LIM?
- :STAT:QUES:LIM:PTR
- :STAT:QUES:LIM:NTR
- :STAT:QUES:LIM:CHAN{1-4}:ENAB
- :STAT:QUES:LIM:CHAN{1-4}:COND?
- :STAT:QUES:LIM:CHAN{1-4}?
- :STAT:QUES:LIM:CHAN{1-4}:PTR
- :STAT:QUES:LIM:CHAN{1-4}:NTR
- :STAT:QUES:BLIM:ENAB
- :STAT:QUES:BLIM:COND?
- :STAT:QUES:BLIM?
- :STAT:QUES:BLIM:PTR
- :STAT:QUES:BLIM:NTR

- :STAT:QUES:BLIM:CHAN{1-4}:ENAB
- :STAT:QUES:BLIM:CHAN{1-4}:COND?
- :STAT:QUES:BLIM:CHAN{1-4}?
- :STAT:QUES:BLIM:CHAN{1-4}:PTR
- :STAT:QUES:BLIM:CHAN{1-4}:NTR
- :STAT:QUES:RLIM:ENAB
- :STAT:QUES:RLIM:COND?
- :STAT:QUES:RLIM?
- :STAT:QUES:RLIM:PTR
- :STAT:QUES:RLIM:NTR
- :STAT:QUES:RLIM:CHAN{1-4}:ENAB
- :STAT:QUES:RLIM:CHAN{1-4}:COND?
- :STAT:QUES:RLIM:CHAN{1-4}?
- :STAT:QUES:RLIM:CHAN{1-4}:PTR
- :STAT:QUES:RLIM:CHAN{1-4}:NTR

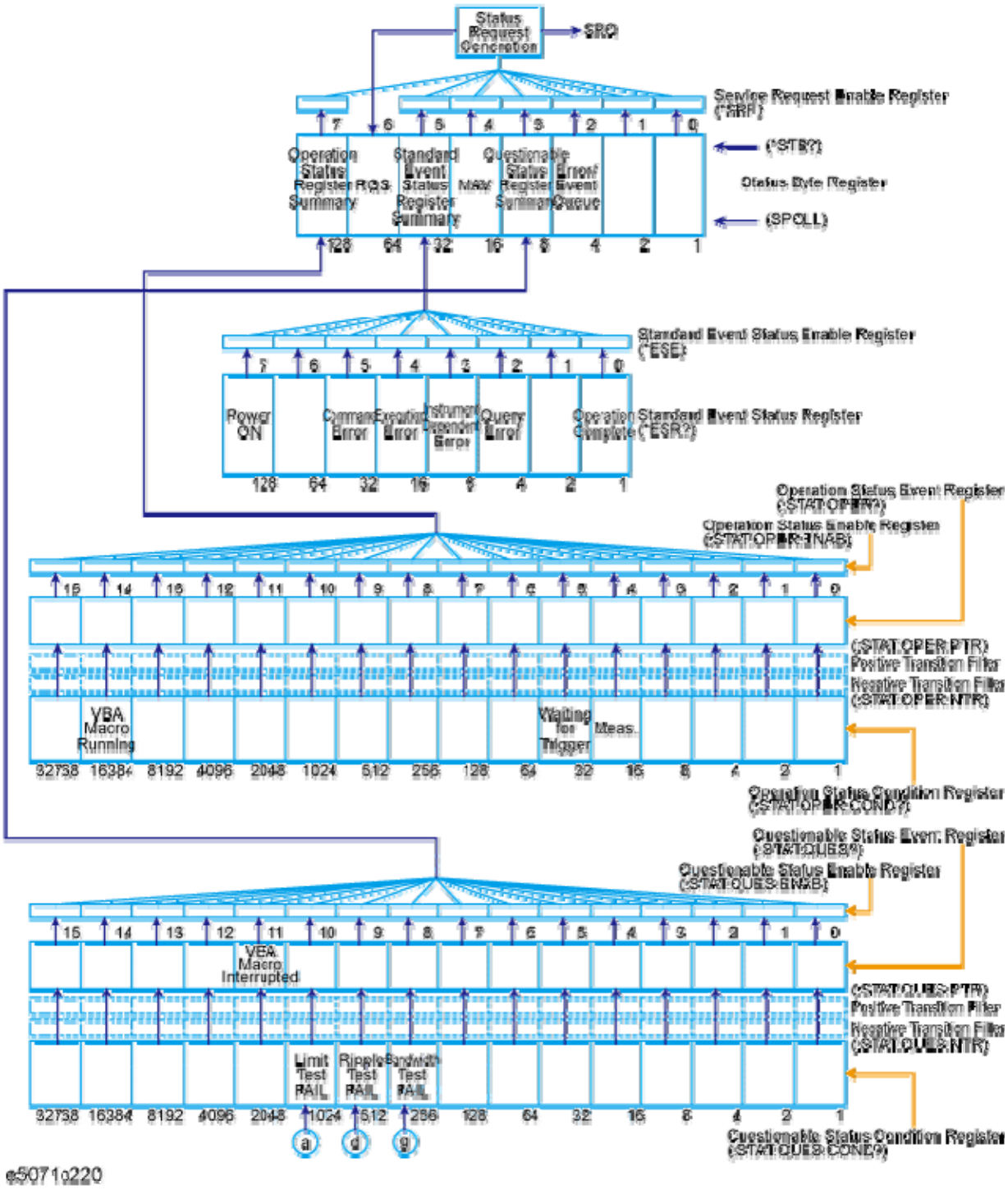
Other topics about Status Reporting System

Status Register Structure

Status Register Structure

- Status Register
- Status Register for Limit Test (Channel)
- Status Register for Limit Test (Trace)
- Status Register for Bandwidth Limit (Channel)
- Status Register for Bandwidth Limit (Trace)
- Status Register for Ripple Limit (Channel)
- Status Register for Ripple Limit (Trace)

Status Register



Status Bit Definitions of Status Byte Register

Bit Position	Name	Description

0, 1	Not used	Always 0
2	Error/Event Queue	Set to "1" if the error/event queue contains data; reset to "0" when all the data has been retrieved.
3	Questionable Status Register Summary	Set to "1" when one of the enabled bits in the questionable status register is set to "1."
4	MAV (Message Available)	Set to "1" when the output queue contains data; reset to "0" when all the data has been retrieved.
5	Standard Event Status Register Summary	Set to "1" when one of the enabled bits in the standard event status register is set to "1."
6	RQS	Set to "1" when any of the status byte register bits enabled by the service request enable register is set to "1"; reset to "0" when all the data has been retrieved through serial polling.
7	Operation Status Register Summary	Set to "1" when one of the enabled bits in the operational status register is set to "1."

Issuing the *CLS command clears all bits from the status byte register.

Status Bit Definitions of Standard Event Status Register

Bit Position	Name	Description
0	Operation Complete	Set to "1" upon completion of all operations done by commands that precede the *OPC? command.
1	Not used	Always 0

2	Query Error	1. Set to "1" when the E5061B receives a data output request but there is no data to output. 2. Set to "1" when the data of the E5061B's output queue has been cleared because of a new message received before the completion of data output.
3	Instrument Dependent Error	Set to "1" when an error has occurred and the error is not a command, query, or execution error.
4	Execution Error	1. Set to "1" when any parameter in an SCPI command exceeds its input range or is inconsistent with the E5061B's capabilities. 2. Set to "1" when an SCPI command cannot be properly executed due to some condition of the E5061B.
5	Command Error	1. Set to "1" when an IEEE 488.2 syntax error occurs (a command sent to the E5061B does not follow the IEEE 488.2 syntax). Possible violations include the command parameter violating the E5061B listening formats or being unacceptable. 2. Set to "1" when a semantic error occurs. Possible causes include a command containing misspellings being sent to the E5061B or an IEEE 488.2 command not supported by the E5061B being sent. 3. Set to "1" when GET (Group Execution Trigger) is input while a program message

		is being received.
6	Not used	Always 0
7	Power ON	Set to "1" when the E5061B is powered ON, or when the firmware is restarted.

Issuing the *CLS command clears all bits from the standard event status register.

Status Bit Definitions of the Operation Status Condition Register

Bit Position	Name	Description
0 - 3	Not used	Always 0
4	Measurement	Set to "1" during measurement
5	Waiting for Trigger	Set to "1" while the instrument is waiting for a trigger.
6 - 13	Not used	Always 0
14	VBA Macro Running	Set to "1" while a VBA macro is running.
15	Not used	Always 0

Issuing the *CLS command clears all bits from the operation status event register.

Status Bit Definitions of the Questionable Status Condition Register

Bit Position	Name	Description
0 - 7	Not used	Always 0
8	Bandwidth Test Fail (Questionable bandwidth limit status register summary)	Set to "1" while one of the enabled bits in the questionable bandwidth limit status event register is set to "1."

9	Ripple Test Fail (Questionable ripple limit status register summary)	Set to "1" while one of the enabled bits in the questionable ripple limit status event register is set to "1."
10	Limit Test Fail (Questionable limit status register summary)	Set to "1" while one of the enabled bits in the questionable limit status event register is set to "1."
11 - 15	Not used	Always 0

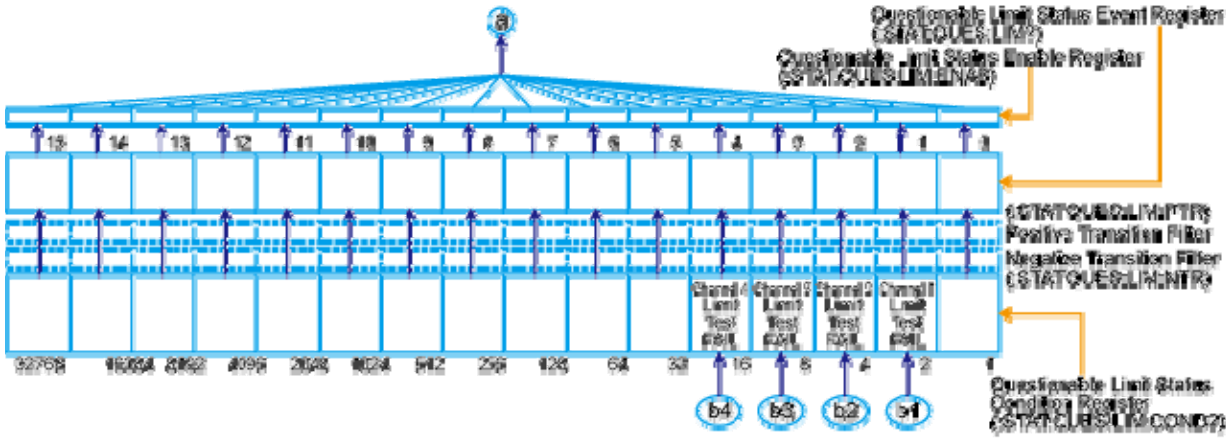
Status Bit Definitions of the Questionable Status Event Register

Bit Position	Name	Description
0 - 7	Not used	Always 0
8	Bandwidth Test Fail (Questionable bandwidth limit status register summary)	Set to "1" when a transition of the condition register occurs if the transition filters are set as valid values.
9	Ripple Test Fail (Questionable ripple limit status register summary)	Set to "1" when a transition of the condition register occurs if the transition filters are set as valid values.
10	Limit Test Fail (Questionable limit status register summary)	Set to "1" when a transition of the condition register occurs if the transition filters are set as valid values.
11	VBA Macro Interrupted	Set to "1" when a VBA macro is interrupted by one of the following

		reasons. Occurrence of an execution error Executing "End" statement in the VBA Macro Executing :PROG:STAT STOP Operating **CTRL** + **Break** using the keyboard Operating **Marcro Break** or **Macro Setup** > **Stop** using the front panel
12 - 15	Not used	Always 0

Issuing the *CLS command clears all bits from the questionable status event register.

Status Register for Limit Test (channel)



e5061b015

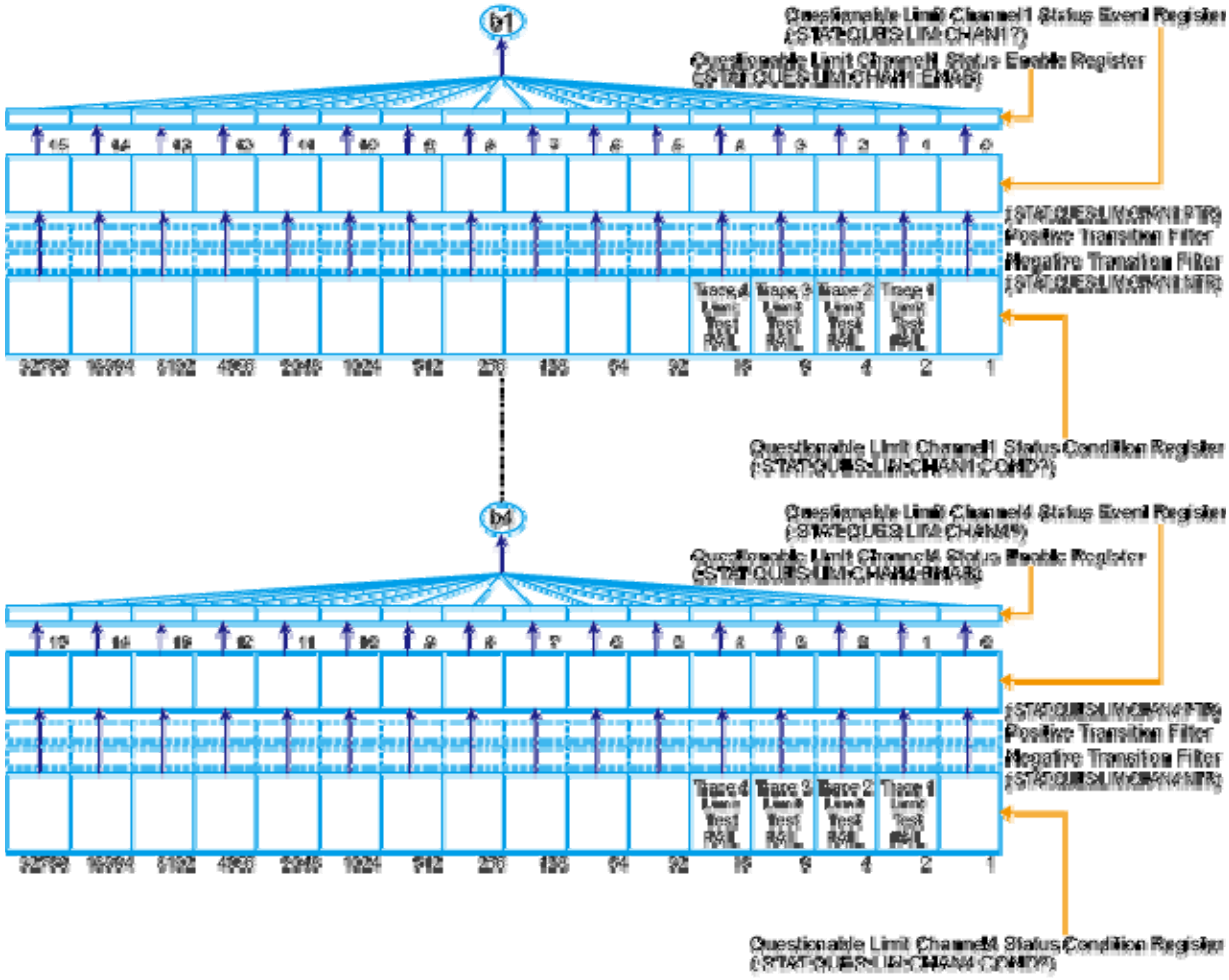
Status Bit Definitions of the Questionable Limit Status Condition Register

Bit Position	Name	Description
0	Not used	Always 0
1	Channel 1 Limit Test Fail (questionable limit channel 1 status register summary)	Set to "1" while one of the enabled bits in the questionable limit channel 1 status event register is set to "1."
2	Channel 2 Limit Test Fail (questionable limit channel 2 status register summary)	Set to "1" while one of the enabled bits in the questionable limit channel 2 status event register is set to "1."
3	Channel 3 Limit Test Fail (questionable limit channel 3 status register summary)	Set to "1" while one of the enabled bits in the questionable limit channel 3 status event register is set to "1."
4	Channel 4 Limit Test Fail (questionable limit channel 4 status register	Set to "1" while one of the enabled bits in the questionable limit channel 4

	summary)	status event register is set to "1."
5 to 15	Not used	Always 0

Issuing the *CLS command clears all bits from the questionable limit status event register.

Status Register for Limit Test (Trace)



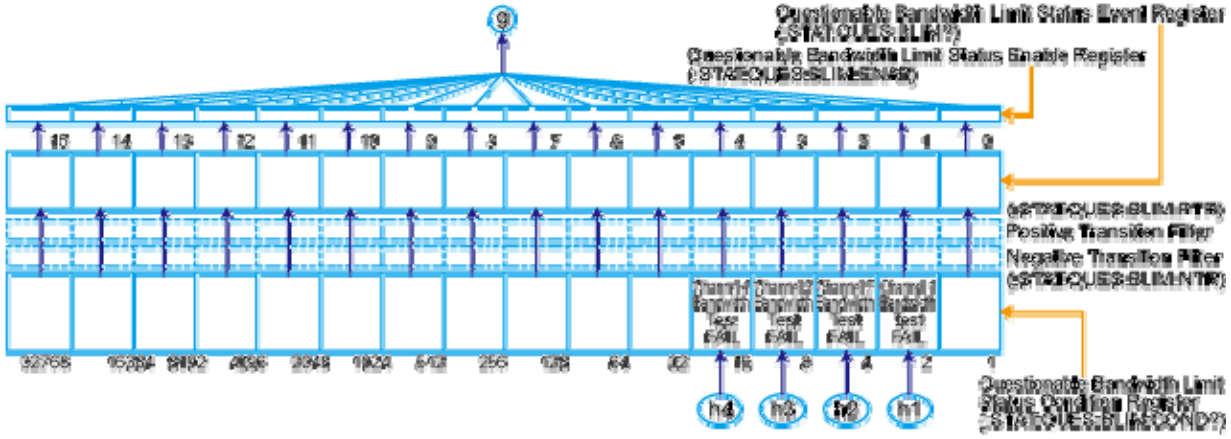
Status Bit Definitions of the Questionable Limit Channel Status Condition Register

Bit Position	Name	Description
0	Not used	Always 0
1	Trace 1 Limit Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the limit test result for trace 1.

2	Trace 2 Limit Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the limit test result for trace 2.
3	Trace 3 Limit Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the limit test result for trace 3.
4	Trace 4 Limit Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the limit test result for trace 4.
5 to 15	Not used	Always 0

Issuing the *CLS command clears all the bits in the questionable limit channel status event register.

Status Register for Bandwidth Limit (Channel)



e5061b017

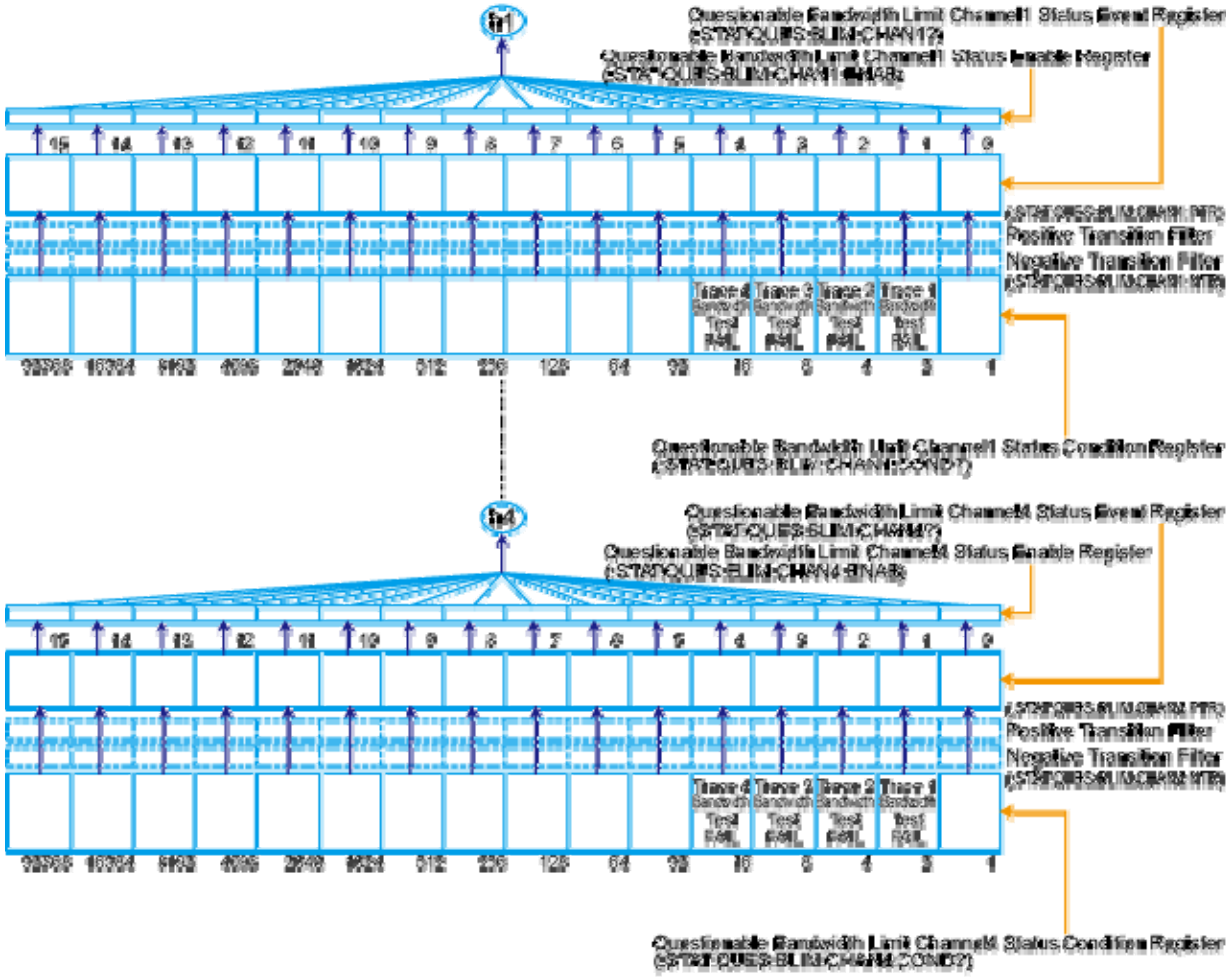
Status Bit Definitions of the Questionable Bandwidth Limit Channel Status Condition Register

Bit Position	Name	Description
0	Not Used	Always 0
1	Channel 1 Bandwidth Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the bandwidth test result for channel 1.
2	Channel 2 Bandwidth Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the bandwidth test result for channel 2.
3	Channel 3 Bandwidth Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the bandwidth test result for channel 3.
4	Channel 4 Bandwidth Test	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle

	Fail	finishes and returns "fail" as the bandwidth test result for channel 4.
5 to 15	Not used	Always 0

Issuing the *CLS command clears all the bits in the questionable bandwidth limit channel {1-4} status event register.

Status Register for Bandwidth Limit (Trace)



e5071b018

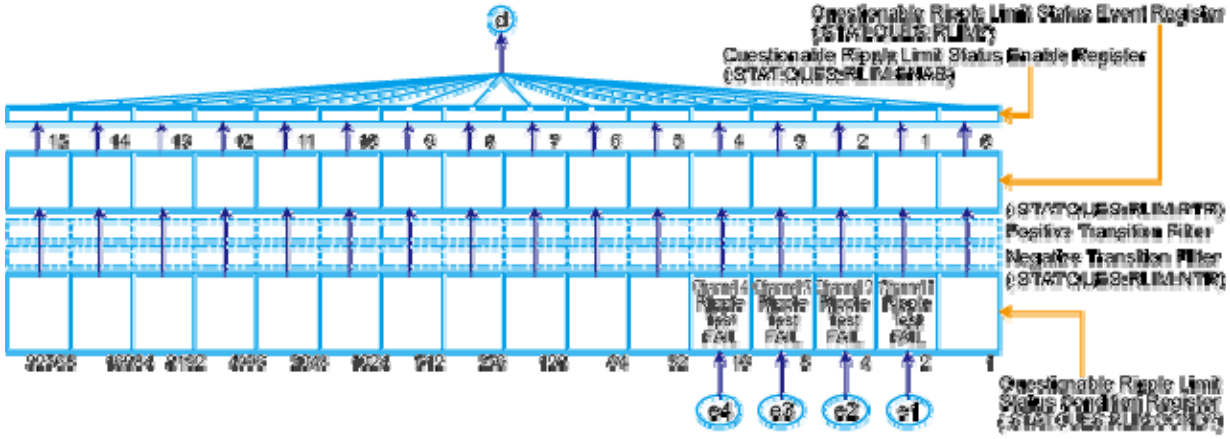
Status Bit Definitions of the Questionable Bandwidth Limit Status Condition Register

Bit Position	Name	Description
0	Not used	Always 0
1	Channel 1 Bandwidth Test Fail (questionable bandwidth limit channel 1 status register summary)	Set to "1" while one of the enabled bits in the questionable bandwidth limit channel 1 status event register is set to "1."

2	Channel 2 Bandwidth Test Fail (questionable bandwidth limit channel 2 status register summary)	Set to "1" while one of the enabled bits in the questionable bandwidth limit channel 2 status event register is set to "1."
3	Channel 3 Bandwidth Test Fail (questionable bandwidth limit channel 3 status register summary)	Set to "1" while one of the enabled bits in the questionable bandwidth limit channel 3 status event register is set to "1."
4	Channel 4 Bandwidth Test Fail (questionable bandwidth limit channel 4 status register summary)	Set to "1" while one of the enabled bits in the questionable bandwidth limit channel 4 status event register is set to "1."
5 to 15	Not used	Always 0

Issuing the *CLS command clears all bits from the questionable bandwidth limit status event register.

Status Register for Ripple Limit (Channel)



e5061b019

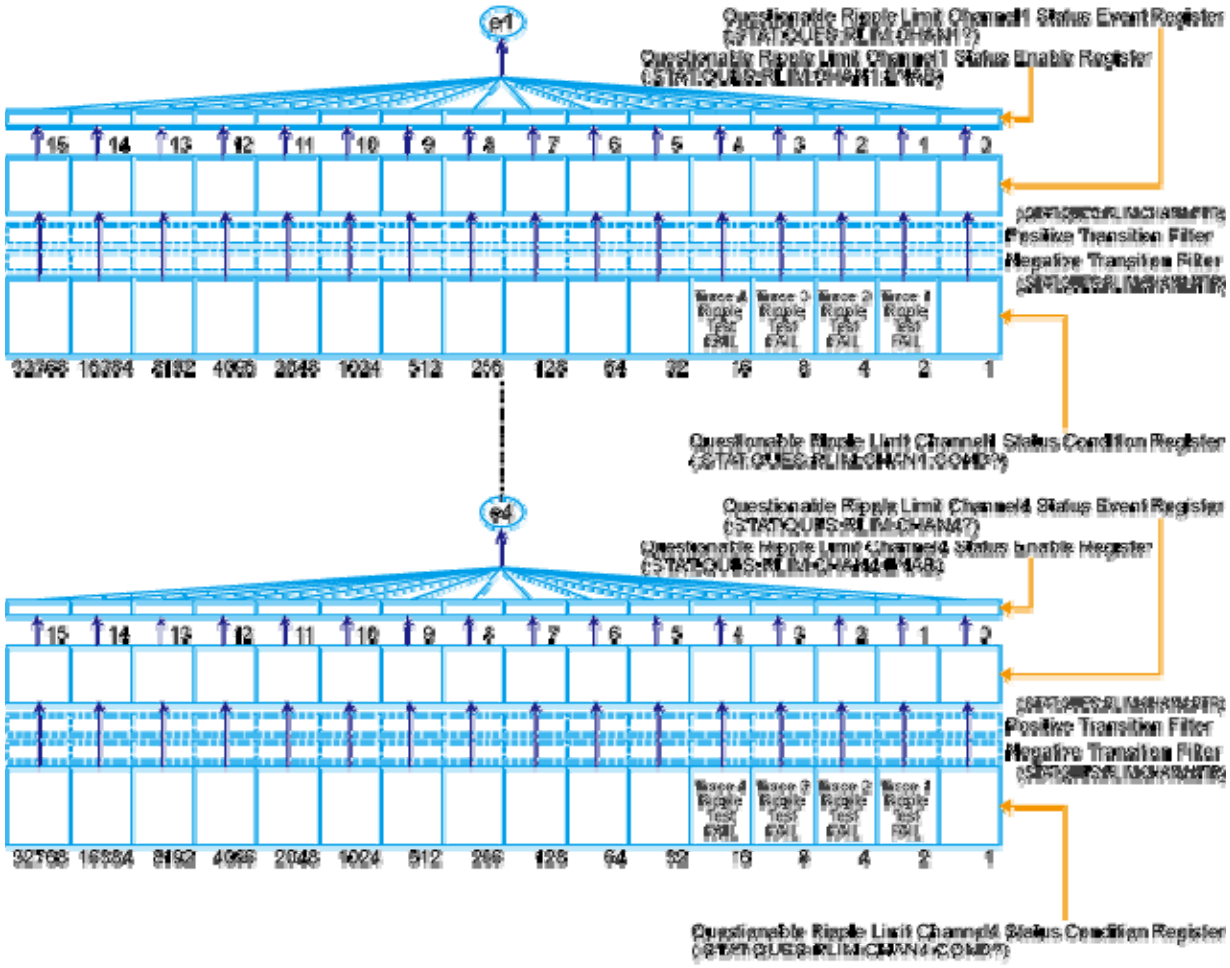
Status Bit Definitions of the Questionable Ripple Limit Status Condition Register

Bit Position	Name	Description
0	Not used	Always 0
1	Channel 1 Ripple Test Fail (questionable ripple limit channel 1 status register summary)	Set to "1" while one of the enabled bits in the questionable ripple limit channel 1 status event register is set to "1."
2	Channel 2 Ripple Test Fail (questionable ripple limit channel 2 status register summary)	Set to "1" while one of the enabled bits in the questionable ripple limit channel 2 status event register is set to "1."
3	Channel 3 Ripple Test Fail (questionable ripple limit channel 3 status register summary)	Set to "1" while one of the enabled bits in the questionable ripple limit channel 3 status event register is set to "1."
4	Channel 4 Ripple Test Fail (questionable ripple limit channel 4 status register	Set to "1" while one of the enabled bits in the questionable ripple limit

	summary)	channel 4 status event register is set to "1."
5 to 15	Not used	Always 0

Issuing the *CLS command clears all bits from the questionable ripple limit status event register.

Status Register for Ripple Limit (Trace)



e5061b020

Status Bit Definitions of the Questionable Ripple Limit Channel Status Condition Register

Bit Position	Name	Description
0	Not used	Always 0
1	Trace 1 Ripple Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the ripple test result for trace 1.

2	Trace 2 Ripple Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the ripple test result for trace 2.
3	Trace 3 Ripple Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the ripple test result for trace 3.
4	Trace 4 Ripple Test Fail	Set to "0" when a measurement cycle begins; set to "1" when the measurement cycle finishes and returns "fail" as the ripple test result for trace 4.
5 to 15	Not used	Always 0

Issuing the *CLS command clears all the bits in the questionable ripple limit channel {1-4} status event register.

Sample Programs

Sample Programs

This section shows sample programs with the SCPI commands which can be executed from the external controller. See Application Programs under VBA Programming about the sample programs for built-in VBA.

- Analyzer Setup
- Calibration
- ECal
- Reading/Writing Error Coefficient
- Waiting for Trigger (OPC?)
- Waiting for Trigger (SRQ)
- Error Detection (SRQ)
- Reading Data in Ascii Format
- Reading Data in Binary Format
- Writing Data in Ascii Format
- Writing Data in Binary Format
- Peak Search
- Bandwidth Search
- Limit Test
- Saving Files
- Transferring Files
- Time Domain
- Control Using SICL-LAN Server
- Controlling Using Telnet Server
- Handler Interface

These sample program files can be downloaded from http://www.agilent.com/find/ena_support.

Analyzer Setup

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

The program listed in this section is a sample program that demonstrates how to configure measurement conditions.

The sample program puts the instrument into the preset state, configures it as shown in table below, and saves the settings to a file named **sample.sta**.

See Setting up Analyzer for this programming.

Target settings

Item		Setting
Window Layout		Channel 1 in the upper window (2/3 of the screen height) and channel 2 in the lower window (1/3 of the screen height)
Channel 1	Sweep type	Segment
	Sweep range	See Segment table.
	Number of measurement points	
	IF bandwidth	
	Power	
	Number of traces	4

	Graph Layout		Four graphs at upper left, upper right, lower left, and lower right.
	Trace 1	Measurement parameter	S11
		Data format	Smith chart (Lin)
		Full-scale value	2
	Trace 2	Measurement parameter	S21
		Data format	Log magnitude
		Reference division line number	9
		Reference division line value	2
		Scale per division	10 dBm
	Trace 3	Measurement parameter	S12
		Data format	Log magnitude
		Reference division line number	9
		Reference division line value	2
		Scale per division	10 dBm
	Trace 4	Measurement parameter	S22
		Data format	Smith chart (Lin)
		Full-scale value	2
Channel	Sweep type		Linear

2	Sweep range	Center value	1.9 GHz
		Span value	500 MHz
	Number of measurement points		101
	IF bandwidth		70 kHz
	Power		0 dBm
	Number of traces		2
	Graph Layout		Two graphs at left and right
	Trace 1	Measurement parameter	S21
		Data format	Log magnitude
		Reference division line number	9
		Reference division line value	2
		Scale per division	10 dBm
	Trace 2	Measurement parameter	S22
		Data format	Smith chart (Lin)
		Full-scale value	2

Segment table for channel 1

Segment Number	Start value	Stop value	Number of measurement points	IF bandwidth	Power
1	1.7 GHz	1.9 GHz	21	50 kHz	0 dBm
2	1.9 GHz	2 GHz	101	10 kHz	-10 dBm
3	2 GHz	2.2 GHz	21	50 kHz	0 dBm

Sample Program in Excel VBA

```
Sub Setup()
```

```
'
Dim defrm As Long
Dim vi As Long
Const TimeOutTime = 20000

Dim Allocate1 As String, Allocate2 As String, File As String
Dim Para1(4) As String, Para2(2) As String
Dim Fmt1(4) As String, Fmt2(2) As String
Dim Star1(3) As String, Stop1(3) As String
Dim IfBw1(3) As Double, IfBw2 As Double
Dim Power1(3) As Double, Power2 As Double
Dim Cent2 As Double, Span2 As Double
Dim RefLev1(4) As Double, RefLev2(2) As Double, Scale1(4) As Double, Scale2(2) As Double
Dim Segm As Integer, Nop1(3) As Integer, Nop2 As Integer
Dim NumOfTr1 As Integer, NumOfTr2 As Integer
Dim RefPos1(4) As Integer, RefPos2(2) As Integer
Dim SendData As String
'
Segm = 3 ' Number of Segment Ch.1 : 3
Star1(1) = "1.7E9"      ' Start Frequency Ch.1 Segm.1: 1.7 GHz
Star1(2) = "1.9E9"      ' Segm.2: 1.9 GHz
Star1(3) = "2E9"        ' Segm.3: 2 GHz
Stop1(1) = "1.9E9"      ' Stop Frequency Ch.1 Segm.1: 1.9 GHz
Stop1(2) = "2E9"        ' Segm.2: 2 GHz
Stop1(3) = "2.2E9"      ' Segm.3: 2.2 GHz
Cent2 = 1900000000#     ' Center Frequency Ch.2 : 1.9 GHz
Span2 = 500000000#      ' Span Ch.2 : 500 MHz
Nop1(1) = 21            ' Number of points Segm.1: 21
Nop1(2) = 101           '               Segm.2: 101
Nop1(3) = 21            '               Segm.3: 21
Nop2 = 101              '       Ch.2 : 101
IfBw1(1) = 50000#       ' IFBW  Ch.1    Segm.1: 50 kHz
IfBw1(2) = 10000#       '               Segm.2: 10 kHz
IfBw1(3) = 50000#       '               Segm.3: 50 kHz
IfBw2 = 70000#          '       Ch.2 : 70 kHz
Power1(1) = 0           ' Power Ch.1    Segm.1: 0 dBm
Power1(2) = -10         '               Segm.2: -10 dBm
```

```
Power1(3) = 0          '              Segm.3: 0 dBm
Power2 = 0             '      Ch.2 : 0 dBm
NumOfTr1 = 4           ' Number of Trace Ch.1 : 4
NumOfTr2 = 2           ' Ch.2 : 2
Allocate1 = "D12_34"   ' Allocate Traces Ch.1 : D12_34
Allocate2 = "D12"      ' Ch.2 : D12
Para1(1) = "S11"       ' Measurement Ch.1 Trace1: S11
Para1(2) = "S21"       ' Parameter Trace2: S21
Para1(3) = "S12"       ' Trace3: S12
Para1(4) = "S22"       ' Trace4: S22
Para2(1) = "S21"       ' Ch.2 Trace1: S21
Para2(2) = "S22"       ' Trace2: S22
Fmt1(1) = "SLIN"       ' Data Format Ch.1 Trace1: Smith(Lin/Phase)
Fmt1(2) = "MLOG"       ' Trace2: Log Mag
Fmt1(3) = "MLOG"       ' Trace3: Log Mag
Fmt1(4) = "SLIN"       ' Trace4: Smith(Lin/Phase)
Fmt2(1) = "MLOG"       ' Ch.2 Trace1: Log Mag
Fmt2(2) = "SLIN"       ' Trace2: Smith(Lin/Phase)
RefPos1(1) = 9         ' Reference Ch.1 Trace2: 9
RefPos1(2) = 9         ' Position Trace3: 9
RefPos2(1) = 9         ' Ch.2 Trace1: 9
RefLev1(1) = 0         ' Reference Level Ch.1 Trace2: 0 dBm
RefLev1(2) = 0         ' Trace3: 0 dBm
RefLev2(1) = 0         ' Ch.2 Trace1: 0 dBm
Scale1(1) = 2          ' Scale Ch.1 Trace1: 2
Scale1(2) = 10         ' Trace2: 10 dBm
Scale1(3) = 10         ' Trace3: 10 dBm
Scale1(4) = 2          ' Trace4: 2
Scale2(1) = 10         ' Ch.2 Trace1: 10 dBm
Scale2(2) = 2          ' Trace2: 2
StaFileName = "sample.sta" ' Save File Name : sample.sta
'
' Assigns a GPIB address to the I/O pass.
Call viOpenDefaultRM(defrm)
Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime) ' Set time out
'
```

```
Call viVPrintf(vi, ":SYST:PRES" + vbLf, 0)
Call viVPrintf(vi, ":DISP:SPL D1_1_2" + vbLf, 0) 'Allocate Channel
Call viVPrintf(vi, ":INIT1:CONT ON" + vbLf, 0) 'Turn on Continuous Activation mode for channel 1
Call viVPrintf(vi, ":INIT2:CONT ON" + vbLf, 0) 'Turn on Continuous Activation mode for channel 2
' ===============
' Setup Channel 1
' ===============
Call viVPrintf(vi, ":SENS1:SWE:TYPE SEGM" + vbLf, 0) 'Sets channel 1 sweep type to segment
'
' Create the data string for Segment Table
SendData = "5,0,1,1,0,0," & Str(Segm)
For i = 1 To Segm
    SendData = SendData + "," & Star1(i) & "," + Stop1(i) & "," & CStr(Nop1(i)) & "," & CStr(IfBw1(i)) &
"," & CStr(Power1(i))
Next i
Call viVPrintf(vi, ":SENS1:SEGM:DATA " + SendData + vbLf, 0)
'
Call viVPrintf(vi, ":CALC1:PAR:COUN " & CStr(NumOfTr1) & vbLf, 0) 'Set number of traces
Call viVPrintf(vi, ":DISP:WIND1:SPL " & Allocate1 & vbLf, 0) 'Set graph layout


For i = 1 To NumOfTr1
    Call viVPrintf(vi, ":CALC1:PAR" & CStr(i) & ":DEF " & Para1(i) & vbLf, 0) 'Set measurement
parameter
    Call viVPrintf(vi, ":CALC1:PAR" & CStr(i) & ":SEL" & vbLf, 0) ' Make trace active
    Call viVPrintf(vi, ":CALC1:FORM " & Fmt1(i) & vbLf, 0) ' Set data format
    '
    Select Case Fmt1(i)
    Case "SLIN", "SLOG", "SCOM", "SMIT", "SADM", "PLIN", "PLOG", "POL"
    ' If data format is neither Smith chart nor polar, sets reference division line number and scale per
division
        Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:PDIV " + CStr(Scale1(i)) + vbLf, 0)
    Case Else
    ' If data format is Smith chart or polar, set full-scale value
        Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:RPOS " & CStr(RefPos1(i)) & vbLf, 0)
        Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:RLEV " & CStr(RefLev1(i)) & vbLf, 0)
        Call viVPrintf(vi, ":DISP:WIND1:TRAC" & CStr(i) & ":Y:PDIV " & CStr(Scale1(i)) & vbLf, 0)
    End Select
Next i
```

```
    ' ===============
    ' Setup Channel 2
    ' ===============
    Call viVPrintf(vi, ":SENS2:SWE:TYPE LIN " + vbLf, 0) ' Set sweep type to linear
    Call viVPrintf(vi, ":SENS2:FREQ:CENT " + CStr(Cent2) + vbLf, 0) ' Set center frequency
    Call viVPrintf(vi, ":SENS2:FREQ:SPAN " + CStr(Span2) + vbLf, 0) ' Set span frequency
    Call viVPrintf(vi, ":SENS2:SWE:POIN " + CStr(Nop2) + vbLf, 0) ' Set number of points
    Call viVPrintf(vi, ":SENS2:BAND " + CStr(IfBw2) + vbLf, 0) ' Set IFBW
    Call viVPrintf(vi, ":SOUR2:POW " + CStr(Power2) + vbLf, 0) ' Set power level
    Call viVPrintf(vi, ":CALC2:PAR:COUN " & CStr(NumOfTr2) & vbLf, 0) ' Set number of traces
    Call viVPrintf(vi, ":DISP:WIND2:SPL " & Allocate2 & vbLf, 0) 'Set graph layout


    For i = 1 To NumOfTr2
        Call viVPrintf(vi, ":CALC2:PAR" & CStr(i) & ":DEF " & Para2(i) & vbLf, 0) 'Set measurement
parameter
        Call viVPrintf(vi, ":CALC2:PAR" & CStr(i) & ":SEL" & vbLf, 0) ' Make trace active
        Call viVPrintf(vi, ":CALC2:FORM " & Fmt2(i) & vbLf, 0) ' Set data format
        '
        Select Case Fmt2(i)
        Case "SLIN", "SLOG", "SCOM", "SMIT", "SADM", "PLIN", "PLOG", "POL"
        ' If data format is neither Smith chart nor polar, sets reference division line number and scale per
division
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:PDIV " + CStr(Scale2(i)) + vbLf, 0)
        Case Else
        ' If data format is Smith chart or polar, set full-scale value
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:RPOS " & CStr(RefPos2(i)) & vbLf, 0)
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:RLEV " & CStr(RefLev2(i)) & vbLf, 0)
            Call viVPrintf(vi, ":DISP:WIND2:TRAC" & CStr(i) & ":Y:PDIV " & CStr(Scale2(i)) & vbLf, 0)
        End Select
    Next i
    '
    Call viVPrintf(vi, ":MMEM:STOR """ & StaFileName & """ & vbLf, 0) ' Save ENA settings to file
    ' Close IO
    Call viClose(vi)
    Call viClose(defrm)

End Sub
```

Sample Program in HT Basic (setup.htb)

```
10 DIM Allocate1$[9],Allocate2$[9],File$[20]
20 DIM Para1$(1:4)[9],Para2$(1:2)[9],Fmt1$(1:4)[9],Fmt2$(1:2)[9]
30 REAL Star1(1:3),Stop1(1:3),Pow1(1:3),Cent2,Span2,Pow2
40 REAL Ref_rev1(1:4),Ref_rev2(1:2),Scale1(1:4),Scale2(1:2)
50 INTEGER Segm,Nop1(1:3),Nop2,Num_of_tr1,Num_of_tr2
60 INTEGER Ref_pos1(1:4),Ref_pos2(1:2),I
70 ASSIGN @Agte506x TO 717
80 !
90 Segm=3 ! Number of Segment Ch.1 : 3
100 Star1(1)=1.7E+9 ! Start Frequency Ch.1 Segm.1: 1.7 GHz
110 Star1(2)=1.9E+9 ! Segm.2: 1.9 GHz
120 Star1(3)=2.E+9 ! Segm.3: 2 GHz
130 Stop1(1)=1.9E+9 ! Stop Frequency Ch.1 Segm.1: 1.9 GHz
140 Stop1(2)=2.E+9 ! Segm.2: 2 GHz
150 Stop1(3)=2.2E+9 ! Segm.3: 2.2 GHz
160 Cent2=1.9E+9 ! Center Frequency Ch.2 : 1.9 GHz
170 Span2=5.00E+8 ! Span Ch.2 : 500 MHz
180 Nop1(1)=21 ! Number Ch.1 Segm.1: 21
190 Nop1(2)=101 ! of Points Segm.2: 101
200 Nop1(3)=21 ! Segm.3: 21
210 Nop2=101 ! Ch.2 : 101
220 If_bw1(1)=5.0E+4 ! IF Bandwidth    Ch.1 Segm.1: 50 kHz
230 If_bw1(2)=1.0E+4 ! Segm.2: 10 kHz
240 If_bw1(3)=5.0E+4 ! Segm.3: 50 kHz
250 If_bw2=7.0E+4 ! Ch.2 : 70 kHz
260 Pow1(1)=0 ! Power Ch.1 Segm.1: 0 dBm
270 Pow1(2)=-10 ! Segm.2: -10 dBm
280 Pow1(3)=0 ! Segm.3: 0 dBm
290 Pow2=0 ! Ch.2 : 0 dBm
300 Num_of_tr1=4 ! Number Ch.1 : 4
310 Num_of_tr2=2 ! of Traces Ch.2 : 2
320 Allocate1$="D12_34" ! Allocate Traces Ch.1 : D12_34
330 Allocate2$="D12" ! Ch.2 : D12
340 Para1$(1)="S11" ! Measurement Ch.1 Trace1: S11
350 Para1$(2)="S21" ! Parameter Trace2: S21
360 Para1$(3)="S12" ! Trace3: S12
370 Para1$(4)="S22" ! Trace4: S22
```

```
380 Para2$(1)="S21" ! Ch.2 Trace1: S21
390 Para2$(2)="S22" ! Trace2: S22
400 Fmt1$(1)="SLIN" ! Data Format Ch.1 Trace1: Smith(Lin/Phase)
410 Fmt1$(2)="MLOG" ! Trace2: Log Mag
420 Fmt1$(3)="MLOG" ! Trace3: Log Mag
430 Fmt1$(4)="SLIN" ! Trace4: Smith(Lin/Phase)
440 Fmt2$(1)="MLOG" ! Ch.2 Trace1: Log Mag
450 Fmt2$(2)="SLIN" ! Trace2: Smith(Lin/Phase)
460 Ref_pos1(2)=9 ! Reference Ch.1 Trace2: 9
470 Ref_pos1(3)=9 ! Position Trace3: 9
480 Ref_pos2(1)=9 ! Ch.2 Trace1: 9
490 Ref_lev1(2)=0 ! Reference Level Ch.1 Trace2: 0 dBm
500 Ref_lev1(3)=0 ! Trace3: 0 dBm
510 Ref_lev2(1)=0 ! Ch.2 Trace1: 0 dBm
520 Scale1(1)=2 ! Scale Ch.1 Trace1: 2
530 Scale1(2)=10 ! Trace2: 10 dBm
540 Scale1(3)=10 ! Trace3: 10 dBm
550 Scale1(4)=2 ! Trace4: 2
560 Scale2(1)=10 ! Ch.2 Trace1: 10 dBm
570 Scale2(2)=2 ! Trace2: 2
580 File$="sample.sta" ! Save File Name : sample.sta
590 !
600 OUTPUT @Agte506x;":SYST:PRES"
610 !
620 OUTPUT @Agte506x;":DISP:SPL D1_1_2"
630 OUTPUT @Agte506x;":INIT1:CONT ON"
640 OUTPUT @Agte506x;":INIT2:CONT ON"
650 !
660 ! Channel 1
670 !
680 OUTPUT @Agte506x;":SENS1:SWE:TYPE SEGM"
690 OUTPUT @Agte506x;":SENS1:SEGM:DATA 5,0,1,1,0,0,";Segm;",";
700 FOR I=1 TO Segm-1
710 OUTPUT @Agte506x;Star1(I);",";Stop1(I);",";Nop1(I);",";If_bw1 (I);",";Pow1(I);",";
720 NEXT I
730 OUTPUT @Agte506x;Star1(Segm);",";Stop1(Segm);",";Nop1(Segm);","
;If_bw1(Segm);",";Pow(Segm)
```

```
740 !
750 OUTPUT @Agte506x;":CALC1:PAR:COUN ";Num_of_tr1
760 OUTPUT @Agte506x;":DISP:WIND1:SPL "&Allocate1$
770 FOR I=1 TO Num_of_tr1
780 OUTPUT @Agte506x;":CALC1:PAR"&VAL$(I)&":DEF "&Para1$(I)
790 OUTPUT @Agte506x;":CALC1:PAR"&VAL$(I)&":SEL"
800 OUTPUT @Agte506x;":CALC1:FORM "&Fmt1$(I)
810 SELECT Fmt1$(I)
820 CASE "SLIN","SLOG","SCOM","SMIT","SADM","PLIN","PLOG","POL"
830 OUTPUT @Agte506x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:PDIV "; Scale1(I)
840 CASE ELSE
850 OUTPUT @Agte506x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:RPOS "; Ref_pos1(I)
860 OUTPUT @Agte506x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:RLEV "; Ref_rev1(I)
870 OUTPUT @Agte506x;":DISP:WIND1:TRAC"&VAL$(I)&":Y:PDIV "; Scale1(I)
880 END SELECT
890 NEXT I
900 !
910 ! Channel 2
920 !
930 OUTPUT @Agte506x;":SENS2:SWE:TYPE LIN"
940 OUTPUT @Agte506x;":SENS2:FREQ:CENT ";Cent2
950 OUTPUT @Agte506x;":SENS2:FREQ:SPAN ";Span2
960 OUTPUT @Agte506x;":SENS2:SWE:POIN ";Nop2
970 OUTPUT @Agte506x;":SENS2:BAND ";If_bw2
980 OUTPUT @Agte506x;":SOUR2:POW ";Pow2
990 !
1000 OUTPUT @Agte506x;":CALC2:PAR:COUN ";Num_of_tr2
1010 OUTPUT @Agte506x;":DISP:WIND2:SPL "&Allocate2$
1020 FOR I=1 TO Num_of_tr2
1030 OUTPUT @Agte506x;":CALC2:PAR"&VAL$(I)&":DEF "&Para2$(I)
1040 OUTPUT @Agte506x;":CALC2:PAR"&VAL$(I)&":SEL"
1050 OUTPUT @Agte506x;":CALC2:FORM "&Fmt2$(I)
1060 SELECT Fmt2$(I)
1070 CASE "SLIN","SLOG","SCOM","SMIT","SADM","PLIN","PLOG","POL"
1080 OUTPUT @Agte506x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:PDIV "; Scale2(I)
1090 CASE ELSE
1100 OUTPUT @Agte506x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:RPOS "; Ref_pos2(I)
```

```
1110 OUTPUT @Agte506x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:RLEV "; Ref_rev2(I)
1120 OUTPUT @Agte506x;":DISP:WIND2:TRAC"&VAL$(I)&":Y:PDIV "; Scale2(I)
1130 END SELECT
1140 NEXT I
1150 !
1160 OUTPUT @Agte506x;":MMEM:STOR """"&File$&""""
1170 END
```

Calibration

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

The sample program performs calibration with the specified calibration type.

See Calibration for this programming.

Sample Program in Excel VBA

```
Sub Cal_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Ch As String
    Dim CalKit As Integer
    Dim Port(2) As String

    Const TimeOutTime = 40000   'timeout time.
    Const Cal85032F = 4         'cal kit number.

    Ch = Cells(5, 5)           'Select channel
    Port(1) = Cells(3, 6)       'Sets the select port 1.
    Port(2) = Cells(3, 7)       'Sets the select port 2.
    CalKit = Cal85032F          'Sets cal kit (85032F)

    Call viOpenDefaultRM(defrm)    'Initializes the VISA system.
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)    'Opens the session to the specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)   'The state of an attribute for the
specified session.

    Call viVPrintf(vi, "*RST" & vbLf, 0)   'Presets the setting state of the ENA.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.

    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:CKIT " & CalKit & vbLf, 0)  'Select the calibration kit

    Select Case Cells(3, 5)
        Case "Response (Open)"   'Perform response calibration (OPEN).
```

```
            Call Cal_Resp(vi, Ch, "OPEN", Port(1))
        Case "Response (Short)"   'Perform response calibration (SHORT).
            Call Cal_Resp(vi, Ch, "Short", Port(1))
        Case "Response (Thru)"    'Perform response calibration (Thru).
            Call Cal_RespThru(vi, Ch, "Thru", Port(1), Port(2))
        Case "Full 1 Port"    'Perform 1-port calibration.
            Call Cal_Slot(vi, Ch, 1, Port)
        Case "Full 2 Port"     'Perform full 2-port calibration.
            Call Cal_Slot(vi, Ch, 2, Port)
    End Select

    Call viClose(vi)  'Closes the resource manager session.
    Call viClose(defrm)  'Breaks the communication and terminates the VISA system.

    End                                                 'End
End Sub

Sub Cal_Resp(vi As Long, Ch As String, CalType As String, Port As String)
    Dim Dummy As Variant   'Variant to receive the result

    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:" & CalType & " " & Port & vbLf, 0)  'Sets the
calibration type.

    MsgBox ("Set " & CalType & " to Port " & Port & ". then click [OK] button")  'Display the message box.

    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:" & CalType & " " & Port & vbLf, 0) 'Measurement the
calibration data.
    Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the *OPC? result.
    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)      'Calculating the calibration
coefficients.

    Call ErrorCheck(vi)      'Checking the error.

End Sub

Sub Cal_RespThru(vi As Long, Ch As String, CalType As String, Port1 As String, Port2 As String)
    Dim Dummy As Variant   'Variant to receive the result.
```

```
    If Port1 <> Port2 Then
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:" & CalType & " " & Port1 & "," & Port2 &
vbLf, 0)   'Sets the calibration type

        MsgBox ("Set " & CalType & " to Port " & Port1 & "&" & Port2 & ". then click [OK] button")   'Display
the message box.

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:" & CalType & " " & Port1 & "," & Port2 & vbLf, 0)
 'Measurement the calibration data.
        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)      'Reads the *OPC? result.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)     'Calculating the calibration
coefficients.

        Call ErrorCheck(vi)     'Checking the error.
    Else
        MsgBox ("Thru calibration select port error!")   'Displaying the error message when selected same
ports.
        Exit Sub
    End If

End Sub
Sub Cal_Slot(vi As Long, Ch As String, NumPort As String, Port() As String)
    Dim Dummy
    Dim i As Integer, j As Integer

    Select Case NumPort
        Case 1
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & vbLf,
0)   'Set the 1-port calibration type.
        Case 2
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & "," &
Port(2) & vbLf, 0)   'Set the full 2-port calibration type.
    End Select
    'Reflection
    For i = 1 To NumPort
        MsgBox ("Set Open to Port " & Port(i) & ". then click [OK] button")   'Display the message box.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:OPEN " & Port(i) & vbLf, 0)    'Measurement the
OPEN calibration.
        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the *OPC? result.
```

```
        MsgBox ("Set Short to Port " & Port(i) & ". then click [OK] button")    'Display the message box.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SHORT " & Port(i) & vbLf, 0)  'Measurement the
SHORT calibration.
        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the *OPC? result.


        MsgBox ("Set Load to Port " & Port(i) & ". then click [OK] button")   'Display the message box.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:LOAD " & Port(i) & vbLf, 0)    'Measurement the
LOAD calibration.
        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)     'Reads the *OPC? result.
    Next i
    'Transmission
    For i = 1 To NumPort - 1
        For j = i + 1 To NumPort
            MsgBox ("Set Thru to Port " & Port(i) & "&" & Port(j) & ". then click [OK] button")   'Display the
message box.
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(i) & "," & Port(j) & vbLf, 0)
  'Measurement the THRU calibration.
            Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the *OPC result.
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(j) & "," & Port(i) & vbLf, 0)
  'Measurement the THRU calibration.
            Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)    'Reads the *OPC result.
        Next j
    Next i
    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)  'Calculating the calibration
coefficients.


    Call ErrorCheck(vi)    'Checking the error.


End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response


    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)    'Reads error message.
    ErrNo = Split(err, ",")    'Gets the error code.


    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)    'Display the message box.
    End If
```

```
End Sub
```

Sample Program in HT Basic (cal.htb)

```
10 DIM File$[20],Ch$[9],Inp_char$[9]
20 INTEGER Cal_kit,Cal_type,Port(1:2)
30 !
40 ASSIGN @Agte506x TO 717
50 File$="Ex_4_1.sta"
60 Ch$="1"
70 !
80 Select_cal_kit(@Agte506x,Ch$)
90 !
100 CLEAR SCREEN
110 ON ERROR GOTO Type_select
120 Type_select: !
130 PRINT "## Calibration Type Selection ##"
140 PRINT " 1: Response (Open)"
150 PRINT " 2: Response (Short)"
160 PRINT " 3: Response (Thru)"
170 PRINT " 4: Full 1 Port"
180 PRINT " 5: Full 2 Port"
210 PRINT ""
220 PRINT "Input 1 to 5"
230 INPUT "Input number? (1 to 5)",Inp_char$
240 Cal_type=IVAL(Inp_char$,10)
250 IF Cal_type<1 OR Cal_type>5 THEN Type_select
260 OFF ERROR
270 !
280 SELECT Cal_type
290 CASE 1
300 Select_port(1,Port(*))
310 Cal_resp(@Agte506x,Ch$,"OPEN",Port(1))
320 CASE 2
330 Select_port(1,Port(*))
340 Cal_resp(@Agte506x,Ch$,"SHOR",Port(1))
350 CASE 3
360 Select_port(2,Port(*))
370 Cal_resp_thru(@Agte506x,Ch$,Port(1),Port(2))
```

```
380 CASE 4
390 Select_port(1,Port(*))
400 Cal_solt(@Agte506x,Ch$,1,Port(*))
410 CASE 5
420 Select_port(2,Port(*))
430 Cal_solt(@Agte506x,Ch$,2,Port(*))
500 END SELECT
510 !
520 OUTPUT @Agte506x;":MMEM:STOR:STYP CST"
530 OUTPUT @Agte506x;":MMEM:STOR """"&File$&""""""
540 END
550 !==========================================
560 ! Calibration Kit Selection Function
570 !==========================================
580 SUB Select_cal_kit(@Agte506x,Ch$)
590 DIM Cal_kit_lbl$(1:10)[20],Inp_char$[9]
600 INTEGER Cal_kit,I
610 CLEAR SCREEN
620 !
630 FOR I=1 TO 10
640 OUTPUT @Agte506x;":SENS1:CORR:COLL:CKIT ";I
650 OUTPUT @Agte506x;":SENS1:CORR:COLL:CKIT:LAB?"
660 ENTER @Agte506x;Cal_kit_lbl$(I)
670 NEXT I
680 ON ERROR GOTO Kit_select
690 Kit_select: !
700 PRINT "## Calibration Kit Selection ##"
710 FOR I=1 TO 10
720 PRINT USING "X,2D,A,X,20A";I,":",Cal_kit_lbl$(I)
730 NEXT I
740 PRINT ""
750 PRINT "Input 1 to 10"
760 INPUT "Input number? (1 to 10)",Inp_char$
770 Cal_kit=IVAL(Inp_char$,10)
780 IF Cal_kit<1 OR Cal_kit>10 THEN Kit_select
790 OFF ERROR
800 !
```

```
810 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:CKIT ";Cal_kit
820 SUBEND
830 !==========================================
840 ! Port Selection Function
850 !==========================================
860 SUB Select_port(INTEGER Num_of_ports,INTEGER Port(*))
870 DIM Inp_char$[9]
880 !
890 CLEAR SCREEN
900 IF Num_of_ports=2 THEN
910 Port(1)=1
920 Port(2)=2
950 ELSE
960 PRINT "## Test Ports Selection ##"
970 ON ERROR GOTO Port_select
990 PRINT "Port(1):";
1000 Port_select:!
1010 INPUT "Number?",Inp_char$
1020 Port(1)=IVAL(Inp_char$,10)
1070 PRINT Port(1)
1090 OFF ERROR
1100 END IF
1110 SUBEND
1120 !==========================================
1130 ! Response (Open/Short) Calibration Function
1140 !==========================================
1150 SUB Cal_resp(@Agte506x,Ch$,Type$,INTEGER Port)
1160 DIM Buff$[9]
1170 !
1180 PRINT "## Response ("&Type$&") Calibration ##"
1190 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:METH:"&Type$&" ";Port
1200 PRINT "Set "&Type$&" to Port "&VAL$(Port)&". Then push [Enter] key."
1210 INPUT "",Buff$
1220 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:"&Type$&" ";Port
1230 OUTPUT @Agte506x;"*OPC?"
1240 ENTER @Agte506x;Buff$
1250 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:SAVE"
```

```
1260 PRINT "Done"
1270 SUBEND
1280 !=========================================
1290 ! Response (Thru) Calibration Function
1300 !=========================================
1310 SUB Cal_resp_thru(@Agte506x,Ch$,INTEGER Port1,Port2)
1320 DIM Buff$[9]
1330 !
1340 PRINT "## Response (Thru) Calibration ##"
1350 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:METH:THRU ";Port1;","; Port2
1360 PRINT "Set THRU between Port "&VAL$(Port1)&" and Port "&VAL$(Port2 )&". Then push [Enter]
key."
1370 INPUT "",Buff$
1380 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:THRU ";Port1;",";Port2
1390 OUTPUT @Agte506x;"*OPC?"
1400 ENTER @Agte506x;Buff$
1410 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:SAVE"
1420 PRINT "Done"
1430 SUBEND
1440 !=========================================
1450 ! Full n Port Calibration Function
1460 !=========================================
1470 SUB Cal_solt(@Agte506x,Ch$,INTEGER Num_of_ports,INTEGER Port(*))
1480 DIM Buff$[9]
1490 INTEGER I,J
1500 !
1510 PRINT "## Full "&VAL$(Num_of_ports)&" Port Calibration ##"
1520 !
1530 ! Calibration Type Selection
1540 !
1550 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:METH:SOLT"&VAL$(Num_of_ ports)&" ";
1560 FOR I=1 TO Num_of_ports-1
1570 OUTPUT @Agte506x;Port(I);",";
1580 NEXT I
1590 OUTPUT @Agte506x;Port(Num_of_ports)
1600 !
1610 ! Reflection Measurement
```

```
1620 !
1630 FOR I=1 TO Num_of_ports
1640 PRINT "Set OPEN to Port "&VAL$(Port(I))&". Then push [Enter] key."
1650 INPUT "",Buff$
1660 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:OPEN ";Port(I)
1670 OUTPUT @Agte506x;"*OPC?"
1680 ENTER @Agte506x;Buff$
1690 PRINT "Set SHORT to Port "&VAL$(Port(I))&". Then push [Enter] key."
1700 INPUT "",Buff$
1710 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:SHOR ";Port(I)
1720 OUTPUT @Agte506x;"*OPC?"
1730 ENTER @Agte506x;Buff$
1740 PRINT "Set LOAD to Port "&VAL$(Port(I))&". Then push [Enter] key."
1750 INPUT "",Buff$
1760 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:LOAD ";Port(I)
1770 OUTPUT @Agte506x;"*OPC?"
1780 ENTER @Agte506x;Buff$
1790 NEXT I
1800 !
1810 ! Transmission Measurement
1820 !
1830 FOR I=1 TO Num_of_ports-1
1840 FOR J=I+1 TO Num_of_ports
1850 PRINT "Set THRU between Port "&VAL$(Port(I))&" and Port "& VAL$(Port(J))&". Then push [Enter]
key."
1860 INPUT "",Buff$
1870 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:THRU ";Port(I);"," ;Port(J)
1880 OUTPUT @Agte506x;"*OPC?"
1890 ENTER @Agte506x;Buff$
1900 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:THRU ";Port(J);"," ;Port(I)
1910 OUTPUT @Agte506x;"*OPC?"
1920 ENTER @Agte506x;Buff$
1930 NEXT J
1940 NEXT I
1950 !
1960 ! Done
1970 !
```

```
1980 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:SAVE"
1990 PRINT "Done"
2000 SUBEND
```

ECal

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

The sample program performs 1-port or 2-port calibration using ECal.

See Calibration for this programming.

Sample Program in Excel VBA

```
Sub ECal_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Ch As String
    Dim CalKit As Integer
    Dim Port(4) As String
    Const TimeOutTime = 40000   'timeout time.

    Ch = Cells(5, 5)           'Select channel
    Port(1) = Cells(3, 6)      'Sets the select port 1.
    Port(2) = Cells(3, 7)      'Sets the select port 2.

    Call viOpenDefaultRM(defrm)     'Initializes the VISA system.
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)      'Opens the session to the specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)   'The state of an attribute for the
specified session.

    Call viVPrintf(vi, "*RST" & vbLf, 0)   'Presets the setting state of the ENA-L.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.

    Select Case Cells(3, 5)
      Case "1 Port"
        Call ECal(vi, Ch, 1, Port)   'Perform 1-port calibration.
      Case "2 Port"
        Call ECal(vi, Ch, 2, Port)   'Perform full 2-port calibration.
    End Select
```

```
    Call viClose(vi)    'Closes the resource manager session.
    Call viClose(defrm)    'Breaks the communication and terminates the VISA system.

    End
End Sub
Sub ECal(vi As Long, Ch As String, NumPort As String, Port() As String)
    Dim Dummy As Variant
    Dim i As Integer, j As Integer

    Select Case NumPort
        Case 1
            MsgBox ("Connect Port " & Port(1) & ". then click [OK] button")  'Display the message box.
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:ECAL:SOLT" & NumPort & " " & Port(1) & vbLf,
0) 'Execute the 1-port calibration.
        Case 2
            MsgBox ("Connect Port " & Port(1) & " and Port " & Port(2) & ". then click [OK] button")   'Display
the message box.
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:ECAL:SOLT" & NumPort & " " & Port(1) & "," &
Port(2) & vbLf, 0) 'Execute the full 2-port calibration.
    End Select

    Call ErrorCheck(vi)      'Checking the error.

End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response

    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)    'Reads error message.
    ErrNo = Split(err, ",")   'Gets the error code.

    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)    'Display the message box.
    End If
End Sub
```

Sample Program in HT Basic (ecal.htb)

```
10 DIM File$[20],Ch$[9],Inp_char$[9]
20 INTEGER Cal_kit,Cal_type,Port(1:2)
30 !
```

```
40 ASSIGN @Agte506x TO 717
50 File$="Ex_4_2.sta"
60 Ch$="1"
70 !
80 CLEAR SCREEN
90 ON ERROR GOTO Type_select
100 Type_select: !
110 PRINT "## Calibration Type Selection ##"
120 PRINT " 1: Full 1 Port"
130 PRINT " 2: Full 2 Port"
160 PRINT ""
170 PRINT "Input 1 to 2"
180 INPUT "Input number? (1 to 2)",Inp_char$
190 Cal_type=IVAL(Inp_char$,10)
200 IF Cal_type<1 OR Cal_type>2 THEN Type_select
210 OFF ERROR
220 !
230 Select_port(Cal_type,Port(*))
240 Ecal(@Agte506x,Ch$,Cal_type,Port(*))
250 !
260 OUTPUT @Agte506x;":MMEM:STOR:STYP CST"
270 OUTPUT @Agte506x;":MMEM:STOR """&File$&"""
280 END
290 !=========================================
300 ! Port Selection Function
310 !=========================================
320 SUB Select_port(INTEGER Num_of_ports,INTEGER Port(*))
330 DIM Inp_char$[9]
340 !
350 CLEAR SCREEN
360 IF Num_of_ports=2 THEN
370 Port(1)=1
380 Port(2)=2
410 ELSE
420 PRINT "## Test Ports Selection ##"
430 ON ERROR GOTO Port_select
450 PRINT "Port(1):";
```

```
460 Port_select: !
470 INPUT "Number?",Inp_char$
480 Port(1)=IVAL(Inp_char$,10)
490 IF Port(1)<1 OR Port(1)>4 THEN Port_select
530 PRINT Port(1)
550 OFF ERROR
560 END IF
570 SUBEND
580 !==========================================
590 ! Electronic Calibration Function
600 !==========================================
610 SUB Ecal(@Agte506x,Ch$,INTEGER Num_of_ports,INTEGER Port(*))
620 DIM Buff$[9],Err_msg$[100]
630 INTEGER Err_no,Port1
640 !
650 PRINT "## Full "&VAL$(Num_of_ports)&" Port ECal ##"
660 !
670 OUTPUT @Agte506x;"*CLS"
680 SELECT Num_of_ports
690 CASE 1
700 PRINT "Connect Port "&VAL$(Port(1))&" to ECal Module."
710 PRINT "Then push [Enter] key."
720 INPUT "",Buff$
730 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:ECAL:SOLT1 ";Port(1)
740 CASE 2
750 PRINT "Connect Port "&VAL$(Port(1));
760 PRINT " and Port "&VAL$(Port(2))&" to ECal Module."
770 PRINT "Then push [Enter] key."
780 INPUT "",Buff$
790 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:ECAL:SOLT2 ";Port(1); ",";Port(2)
920 END SELECT
930 PRINT "Executing ..."
940 OUTPUT @Agte506x;":SYST:ERR?"
950 ENTER @Agte506x;Err_no,Err_msg$
960 IF Err_no<>0 THEN
970 PRINT "Error occurred!!"
980 PRINT " No:";Err_no,"Description: "&Err_msg$
```

```
990 PRINT "ECAL INTERRUPT!!"
1000 ELSE
1010 PRINT "Done"
1020 END IF
1030 SUBEND
```

Reading / Writing Error Coefficient

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program reads/writes the error coefficient.

This program sets measurement conditions and perform full 2-port calibration, preset the E5061B with the read error coefficient to be written, and then again read the error coefficient.

NOTE The error coefficient read from the E5061B is displayed in a graph.

Sample Program in Excel VBA

```
Sub Err_Term_Click()
    Dim defrm As Long          'Session to Default Resource Manager
    Dim vi As Long             'Session to instrument
    Dim Ch As String
    Dim CalKit As Integer
    Dim Port(2) As String
    Dim Result As String * 10
    Dim tNop As Long
    Dim Respons As String
    Dim Stimulus As String
    Dim ErrTerm As String

    Const TimeOutTime = 40000   'timeout time.
    Const Cal85032F = 4         'cal kit number

    Ch = Cells(2, 6)           'Select channel
    Port(1) = Cells(4, 6)       'Sets the select port 1.
    Port(2) = Cells(5, 6)       'Sets the select port 2.
    Respons = Cells(6, 6)       'Sets the respons port.
    Stimulus = Cells(7, 6)      'Sets the stimulus port.
    ErrTerm = Cells(8, 6)       'Sets the error term.

    CalKit = Cal85032F          'Set cal kit (85032F)
```

```
    Call viOpenDefaultRM(defrm)    'Initializes the VISA system.
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)    'Opens the session to the specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)      'The state of an attribute for the
specified session.

    Call viVPrintf(vi, "*RST" & vbLf, 0)   'Presets the setting state of the ENA.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.

    Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:CKIT " & CalKit & vbLf, 0)  'Select the calibration kit.

    Call Set_sgm_tbl(vi, Ch)     'Configures the segment table.

    Select Case Cells(3, 6)    'Sets the read/write.
       Case "Read"
          Call Cal_Slot(vi, Ch, 2, Port)  'Full 2-Port Calibration.
       Case "Write"
          Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF:METH:SOLT2 1,2" & vbLf, 0)   'Sets the
calibration type to the full 2-port calibration.
    End Select

    Call viVPrintf(vi, ":SENS" & Ch & ":SEGM:SWE:POIN?" & vbLf, 0)  'Reads out the total number of the
measurement points of all segments.
    Call viVScanf(vi, "%t", Result)

    Call Exec_Error_Term(vi, Ch, Val(Result), ErrTerm, Respons, Stimulus)    'Reads the error coefficient.

    Call viClose(vi)      'Closes the resource manager session.
    Call viClose(defrm)      'Breaks the communication and terminates the VISA system.

    End
End Sub
Sub Exec_Error_Term(vi As Long, Ch As String, Nop As Long, ErrTerm As String, Respons As String,
Stimulus As String)

    Dim Error_Term_Data As Variant
    Dim Freq_Data As Variant
    Dim i As Integer, j As Integer
```

```
    Dim SelMode As String
    Dim Result As String * 10000
    Dim RealData As Double
    Dim ImagData As Double
    Dim FreqData As Double

    ReDim Error_Term_Data(Nop * 2) As String        'Defines the stock variables for the error coefficient
as needed for NOP.
    ReDim Freq_Data(Nop) As String                  'Defines the stock variables for the frequency values.

    SelMode = Cells(3, 6)   'Reads the read/write mode.

    Select Case SelMode
        Case "Read"                             'Reads the error coefficient from the ena.
            Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF? " & ErrTerm & "," & Respons & "," & Stimulus &
vbLf, 0)  'Read the calibration coefficient data.
            Call viVScanf(vi, "%t", Result)
            Error_Term_Data = Split(Result, ",")    'Splits the read data by comma.

            Freq_Data = Make_Freq(vi, Nop)      'Calculates the frequency values.

            For i = 0 To Nop - 1
                RealData = CDbl(Error_Term_Data(i * 2))       'Reads the real data from error coefficient items.
                ImagData = CDbl(Error_Term_Data(i * 2 + 1))  'Reads the imag data from error coefficient
items.
                FreqData = CDbl(Freq_Data(i + 1))             'Reads the frequency values.
                Cells(10 + i, 2) = RealData                   'Displays the real data to the excel sheet.
                Cells(10 + i, 3) = ImagData                   'Displays the imag data to the excel sheet.
                Cells(10 + i, 1) = FreqData                   'Displays the frequency values to the excel sheet.
            Next i

        Case "Write"     'Write the error coefficient to the ena.
            Error_Term_Data = ErrTerm & "," & Respons & "," & Stimulus     'Sets the command parameter.
            For i = 0 To Nop - 1
                RealData = Cells(10 + i, 2)                   'Retrieves the real data from the excel sheet.
                ImagData = Cells(10 + i, 3)                   'Retrieves the imag data from the excel sheet.
                Error_Term_Data = Error_Term_Data & "," & RealData & "," & ImagData         'Sets the
command parameter.
```

```
        Next i

        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF " & Error_Term_Data & vbLf, 0)    'Write the
calibration coefficient data.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COEF:SAVE" & vbLf, 0)         'Calculates the
calibration coefficients.

    End Select
End Sub
Function Make_Freq(vi As Long, tPoint As Long) As Variant
    Dim start_freq As Double
    Dim stop_freq As Double
    Dim Nop As Integer
    Dim fStep As Double
    Dim fPoint As Double
    Dim freq_arry() As Variant
    Dim MeasPoint As Integer

    Const SegmentCnt = 2        'number of segment table.

    ReDim freq_arry(tPoint) As Variant

    MeasPoint = 1

    For j = 1 To SegmentCnt
        start_freq = Cells(3 + j - 1, 9)                    'Sets the start frequency of segment table.
        stop_freq = Cells(3 + j - 1, 10)                    'Sets the stop frequency of segment table.
        Nop = Cells(3 + j - 1, 13)                          'Sets the nop of segment table.
        fStep = (stop_freq - start_freq) / (Nop - 1)        'Calculate the frequency step.

        fPoint = start_freq                                 'Sets the frequency start point.
        For i = 1 To Nop
            freq_arry(MeasPoint) = fPoint                   'Sets the frequency value.
            fPoint = fPoint + fStep                         'Calculate the frequency points.
            MeasPoint = MeasPoint + 1                       'Add to measurement points.
        Next i
    Next j
```

```
    Make_Freq = freq_arry                                   'Sets the frequency data array.

End Function

Sub Set_sgm_tbl(vi As Long, Ch As String)
    Dim Star1(2) As Double, Stop1(2) As Double, Pow1(2) As Double, If_bw1(2) As Double
    Dim Segm As Integer, Nop1(2) As Integer, Num_of_tr1 As Integer
    Dim i As Integer

    Segm = 2
    Star1(1) = Cells(3, 9)              'Sets the start frequency of segment 1 table.
    Stop1(1) = Cells(3, 10)              'Sets the stop frequency of segment 1 table.
    Pow1(1) = Cells(3, 11)               'Sets the power of segment 1 table.
    If_bw1(1) = Cells(3, 12)             'Sets the ifbw of segment 1 table.
    Nop1(1) = Cells(3, 13)               'Sets the nop of segment 1 table.
    Star1(2) = Cells(4, 9)              'Sets the start frequency of segment 2 table.
    Stop1(2) = Cells(4, 10)              'Sets the stop frequency of segment 2 table.
    Pow1(2) = Cells(4, 11)               'Sets the power of segment 2 table.
    If_bw1(2) = Cells(4, 12)             'Sets the ifbw of segment 2 table.
    Nop1(2) = Cells(4, 13)               'Sets the nop of segment 2 table.

    Call viVPrintf(vi, ":SENS" & Ch & ":SWE:TYPE SEGM" & vbLf, 0)   'Sets sweep type to segment.
    Call viVPrintf(vi, ":SENS" & Ch & ":SEGM:DATA 5,0,1,1,0,0," & Segm & ",", 0)       'Sets the header of
segment table.
    Call viVPrintf(vi, Star1(1) & "," & Stop1(1) & "," & Nop1(1) & "," & If_bw1(1) & "," & Pow1(1) & ",", 0)
 'Sets the 1st parameter.
    Call viVPrintf(vi, Star1(2) & "," & Stop1(2) & "," & Nop1(2) & "," & If_bw1(2) & "," & Pow1(2) & vbLf, 0)
 'Sets the 2nd parameter.

    Call ErrorCheck(vi)   'Checking the error.

End Sub
Sub Cal_Slot(vi As Long, Ch As String, NumPort As String, Port() As String)
    Dim Dummy
    Dim i As Integer, j As Integer

    Select Case NumPort
        Case 1
```

```
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & vbLf, 0)   'Set the 1-port calibration type.
      Case 2
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:METH:SOLT" & NumPort & " " & Port(1) & "," & Port(2) & vbLf, 0)  'Set the full 2-port calibration type.
   End Select
   'Reflection
   For i = 1 To NumPort
      MsgBox ("Set Open to Port " & Port(i) & ". then click [OK] button")    'Display the message box.
      Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:OPEN " & Port(i) & vbLf, 0)      'Measurement the OPEN calibration.
      Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)     'Reads the *OPC? result.

      MsgBox ("Set Short to Port " & Port(i) & ". then click [OK] button")  'Display the message box.
      Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SHORT " & Port(i) & vbLf, 0)  'Measurement the SHORT calibration.
      Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)  'Reads the *OPC? result.

      MsgBox ("Set Load to Port " & Port(i) & ". then click [OK] button")   'Display the message box.
      Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:LOAD " & Port(i) & vbLf, 0)   'Measurement the LOAD calibration.
      Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)   'Reads the *OPC? result.
   Next i
   'Transmission
   For i = 1 To NumPort - 1
      For j = i + 1 To NumPort
        MsgBox ("Set Thru to Port " & Port(i) & "&" & Port(j) & ". then click [OK] button")  'Display the message box.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(i) & "," & Port(j) & vbLf, 0)
 'Measurement the THRU calibration.
        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)   'Reads the *OPC? result.
        Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:THRU " & Port(j) & "," & Port(i) & vbLf, 0)
 'Measurement the THRU calibration.
        Call viVQueryf(vi, "*OPC?" & vbLf, "%t", Dummy)      'Reads the *OPC? result.
      Next j
   Next i

   Call viVPrintf(vi, ":SENS" & Ch & ":CORR:COLL:SAVE" & vbLf, 0)    'Calculating the calibration coefficients.
```

```
    Call ErrorCheck(vi)   'Checking the error.

End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response

    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)   'Reads error message.
    ErrNo = Split(err, ",")      'Gets the error code.

    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)    'Display the message box.
    End If
End Sub
```

Sample Program in HT Basic (ErrTerm.htb)

```
2000 Main:!
2010 INTEGER Agte506x,Ii,Nop
2020 INTEGER Respons,Stimulas
2030 INTEGER Port(1:2)
2040 REAL Stok(12,1:5000)
2050 REAL Stok2(12,1:5000)
2060 REAL Stok3(12,1:5000)
2070 DIM Ch$[10],Wk$[128]
2080 !
2090 ! PC's Monitor Clear
2100 CLEAR SCREEN
2110 GINIT
2130 !
2140 ! Set ENA++'s Addr
2150 Agte506x=717
2160 !
2170 Ch$="1"
2180 !
2190 ! Set ENA++'s I/O Path
2200 ASSIGN @Agte506x TO Agte506x
2210 !
2220 ON TIMEOUT SC(@Agte506x),15 RECOVER Tout
2230 !
```

```
2240 ! Set Start Port and End Port
2250 Port(1)=1
2260 Port(2)=2
2270 !
2280 ! Setup Segment Table
2290 CALL Set_sgm_tbl(@Agte506x)
2300 !
2310 ! Select Cal Kit
2320 CALL Select_cal_kit(@Agte506x,Ch$)
2330 !
2340 ! Execute Full-2Port Calibration
2350 CALL Cal_solt(@Agte506x,Ch$,2,Port(*))
2360 !
2370 ! Get All Segment's Points
2380 CALL Get_nop(@Agte506x,Nop,Ch$)
2390 !
2400 REDIM Stok(12,1:Nop*2)
2410 REDIM Stok2(12,1:Nop*2)
2420 REDIM Stok3(12,1:Nop*2)
2430 !
2440 CALL Exec_error_term(@Agte506x,"READ","ES",Ch$,1,Nop,1,1,Stok(*))
2450 CALL Exec_error_term(@Agte506x,"READ","ES",Ch$,2,Nop,2,2,Stok(*))
2460 CALL Exec_error_term(@Agte506x,"READ","ER",Ch$,3,Nop,1,1,Stok(*))
2470 CALL Exec_error_term(@Agte506x,"READ","ER",Ch$,4,Nop,2,2,Stok(*))
2480 CALL Exec_error_term(@Agte506x,"READ","ED",Ch$,5,Nop,1,1,Stok(*))
2490 CALL Exec_error_term(@Agte506x,"READ","ED",Ch$,6,Nop,2,2,Stok(*))
2500 !
2510 CALL Exec_error_term(@Agte506x,"READ","EL",Ch$,7,Nop,1,2,Stok(*))
2520 CALL Exec_error_term(@Agte506x,"READ","EL",Ch$,8,Nop,2,1,Stok(*))
2530 CALL Exec_error_term(@Agte506x,"READ","ET",Ch$,9,Nop,1,2,Stok(*))
2540 CALL Exec_error_term(@Agte506x,"READ","ET",Ch$,10,Nop,2,1,Stok(*))
2550 !
2560 CLEAR SCREEN
2570 PRINT "Push [Preset] - OK of ENA. Then push [Enter] key."
2580 INPUT "",Wk$
2590 !
2600 CALL Set_sgm_tbl(@Agte506x)
```

```
2610 !
2620 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COEF:METH:SOLT2 ";Port(1);",";Port(2)
2630 !
2640 CALL Exec_error_term(@Agte506x,"WRITE","ES",Ch$,1,Nop,1,1,Stok(*))
2650 CALL Exec_error_term(@Agte506x,"WRITE","ES",Ch$,2,Nop,2,2,Stok(*))
2660 CALL Exec_error_term(@Agte506x,"WRITE","ER",Ch$,3,Nop,1,1,Stok(*))
2670 CALL Exec_error_term(@Agte506x,"WRITE","ER",Ch$,4,Nop,2,2,Stok(*))
2680 CALL Exec_error_term(@Agte506x,"WRITE","ED",Ch$,5,Nop,1,1,Stok(*))
2690 CALL Exec_error_term(@Agte506x,"WRITE","ED",Ch$,6,Nop,2,2,Stok(*))
2700 !
2710 CALL Exec_error_term(@Agte506x,"WRITE","EL",Ch$,7,Nop,1,2,Stok(*))
2720 CALL Exec_error_term(@Agte506x,"WRITE","EL",Ch$,8,Nop,2,1,Stok(*))
2730 CALL Exec_error_term(@Agte506x,"WRITE","ET",Ch$,9,Nop,1,2,Stok(*))
2740 CALL Exec_error_term(@Agte506x,"WRITE","ET",Ch$,10,Nop,2,1,Stok(*))
2750 !
2760 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COEF:SAVE"
2770 !
2780 CALL Exec_error_term(@Agte506x,"READ","ES",Ch$,1,Nop,1,1,Stok2(*))
2790 CALL Exec_error_term(@Agte506x,"READ","ES",Ch$,2,Nop,2,2,Stok2(*))
2800 CALL Exec_error_term(@Agte506x,"READ","ER",Ch$,3,Nop,1,1,Stok2(*))
2810 CALL Exec_error_term(@Agte506x,"READ","ER",Ch$,4,Nop,2,2,Stok2(*))
2820 CALL Exec_error_term(@Agte506x,"READ","ED",Ch$,5,Nop,1,1,Stok2(*))
2830 CALL Exec_error_term(@Agte506x,"READ","ED",Ch$,6,Nop,2,2,Stok2(*))
2840 !
2850 CALL Exec_error_term(@Agte506x,"READ","EL",Ch$,7,Nop,1,2,Stok2(*))
2860 CALL Exec_error_term(@Agte506x,"READ","EL",Ch$,8,Nop,2,1,Stok2(*))
2870 CALL Exec_error_term(@Agte506x,"READ","ET",Ch$,9,Nop,1,2,Stok2(*))
2880 CALL Exec_error_term(@Agte506x,"READ","ET",Ch$,10,Nop,2,1,Stok2(*))
2890 !
2900 ASSIGN @Agte506x TO *
2910 !
2920 DISP CHR$(139)&" Done ..."&CHR$(136)
2930 STOP
2940 !
2950 Tout: OFF TIMEOUT SC(@Agte506x)
2960 !
2970 ASSIGN @Agte506x TO *
```

```
2980 !
2990 PRINT CHR$(137)&" ENA Timeout ..."&CHR$(136)
3000 END
3010!
3020 Set_sgm_tbl: SUB Set_sgm_tbl(@Agte506x)
3030 REAL Star1(1:2),Stop1(1:2),Pow1(1:2)
3040 INTEGER Segm,Nop1(1:2),Num_of_tr1
3050 INTEGER I
3060 !
3070 CLEAR SCREEN
3080 DISP CHR$(138)&" Wait ..."&CHR$(136)
3090 !
3100 Segm=2 ! Number of Segment Ch.1 : 2
3110 Star1(1)=3.E+6 ! Start Frequency Ch.1 Segm.1: 3.0 MHz
3120 Star1(2)=5.0E+7 ! Segm.2: 50.0 MHz
3130 Stop1(1)=1.0E+7 ! Stop Frequency Ch.1 Segm.1: 10.0 MHz
3140 Stop1(2)=8.E+9 ! Segm.2: 8.0 GHz
3150 Nop1(1)=2 ! Number Ch.1 Segm.1: 2
3160 Nop1(2)=170 ! of Points Segm.2: 170
3170 If_bw1(1)=7.0E+4 ! IF Bandwidth Ch.1 Segm.1: 70 kHz
3180 If_bw1(2)=7.0E+4 ! Segm.2: 70 kHz
3190 Pow1(1)=0 ! Power Ch.1 Segm.1: 0 dBm
3200 Pow1(2)=0 ! Segm.2: 0 dBm
3210 !
3220 OUTPUT @Agte506x;":SYST:PRES"
3230 !
3240 WAIT 5
3250 !
3260 ! Channel 1
3270 !
3280 OUTPUT @Agte506x;":SENS1:SWE:TYPE SEGM"
3290 OUTPUT @Agte506x;":SENS1:SEGM:DATA 5,0,1,1,0,0,";Segm;",";
3300 FOR I=1 TO Segm-1
3310 OUTPUT @Agte506x;Star1(I);",";Stop1(I);",";Nop1(I);",";If_bw1(I);",";Pow1(I);",";
3320 NEXT I
3330 OUTPUT
@Agte506x;Star1(Segm);",";Stop1(Segm);",";Nop1(Segm);",";If_bw1(Segm);",";Pow1(Segm)
```

```
3340 !
3350 OUTPUT @Agte506x;":CALC1:PAR:COUN ";Num_of_tr1
3360 FOR I=1 TO Num_of_tr1
3370 OUTPUT @Agte506x;":CALC1:PAR"&VAL$(I)&":SEL"
3380 NEXT I
3390 SUBEND
3400!
3410 Select_cal_kit: SUB Select_cal_kit(@Agte506x,Ch$)
3420 !=========================================
3430 ! Calibration Kit Selection Function
3440 !=========================================
3450 !
3460 DIM Cal_kit_lbl$(1:10)[20],Inp_char$[9]
3470 DIM Msg$[80],Wk$[10]
3480 INTEGER Cal_kit,I,Noc
3490 !
3500 ! PC's Monitor Clear
3510 CLEAR SCREEN
3520 !
3530 ! Number of Cal Kid
3540 Noc=10
3550 !
3560 FOR I=1 TO Noc
3570 OUTPUT @Agte506x;":SENS1:CORR:COLL:CKIT ";I
3580 OUTPUT @Agte506x;":SENS1:CORR:COLL:CKIT:LAB?"
3590 ENTER @Agte506x;Cal_kit_lbl$(I)
3600 NEXT I
3610 ON ERROR GOTO Kit_select
3620 !
3630 PRINT "## Calibration Kit Selection ##"
3640 FOR I=1 TO Noc
3650 PRINT USING "X,2D,A,X,20A";I,":",Cal_kit_lbl$(I)
3660 NEXT I
3670 PRINT ""
3680 PRINT "Input 1 to "&VAL$(Noc)
3690 !
3700 Msg$="Input number? (1 to "&VAL$(Noc)&") "
```

```
3710 LOOP
3720 LOOP
3730 DISP Msg$;
3740 INPUT Inp_char$
3750 Cal_kit=IVAL(Inp_char$,10)
3760 EXIT IF 1<=Cal_kit AND Cal_kit<=Noc
3770 Kit_select:!
3780 BEEP
3790 END LOOP
3800 !
3810 Wk$=""
3820 PRINT TABXY(1,Cal_kit+1);
3830 PRINT USING "X,B,2D,A,X,20A,B";139,Cal_kit,":",Cal_kit_lbl$(Cal_kit),136
3840 INPUT "Sure ? [Y/N]",Wk$
3850 EXIT IF (UPC$(Wk$)="Y")
3860 PRINT TABXY(1,Cal_kit+1);
3870 PRINT USING "X,2D,A,X,20A";Cal_kit,":",Cal_kit_lbl$(Cal_kit)
3880 BEEP
3890 BEEP
3900 END LOOP
3910 OFF ERROR
3920 !
3930 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:CKIT ";Cal_kit
3940 SUBEND
3950!
3960 Cal_solt: SUB Cal_solt(@Agte506x,Ch$,INTEGER Num_of_ports,INTEGER Port(*))
3970 !=========================================
3980 ! Full n Port Calibration Function
3990 !=========================================
4000 !
4010 DIM Buff$[9]
4020 INTEGER I,J
4030 !
4040 ! PC's Monitor Clear
4050 CLEAR SCREEN
4060 !
4070 PRINT "## Full "&VAL$(Num_of_ports)&" Port Calibration ##"
```

```
4080 !
4090 ! Calibration Type Selection
4100 !
4110 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:METH:SOLT"&VAL$(Num_of_ports)&" ";
4120 FOR I=1 TO Num_of_ports-1
4130 OUTPUT @Agte506x;Port(I);",";
4140 NEXT I
4150 OUTPUT @Agte506x;Port(Num_of_ports)
4160 !
4170 ! Reflection Measurement
4180 !
4190 FOR I=1 TO Num_of_ports
4200 PRINT "Set OPEN to Port "&VAL$(Port(I))&". Then push [Enter] key."
4210 INPUT "",Buff$
4220 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:OPEN ";Port(I)
4230 OUTPUT @Agte506x;"*OPC?"
4240 ENTER @Agte506x;Buff$
4250 PRINT "Set SHORT to Port "&VAL$(Port(I))&". Then push [Enter] key."
4260 INPUT "",Buff$
4270 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:SHOR ";Port(I)
4280 OUTPUT @Agte506x;"*OPC?"
4290 ENTER @Agte506x;Buff$
4300 PRINT "Set LOAD to Port "&VAL$(Port(I))&". Then push [Enter] key."
4310 INPUT "",Buff$
4320 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:LOAD ";Port(I)
4330 OUTPUT @Agte506x;"*OPC?"
4340 ENTER @Agte506x;Buff$
4350 NEXT I
4360 !
4370 ! Transmission Measurement
4380 !
4390 FOR I=1 TO Num_of_ports-1
4400 FOR J=I+1 TO Num_of_ports
4410 PRINT "Set THRU between Port "&VAL$(Port(I))&" and Port "&VAL$(Port(J))&". Then push [Enter]
key."
4420 INPUT "",Buff$
4430 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:THRU ";Port(I);",";Port(J)
```

```
4440 OUTPUT @Agte506x;"*OPC?"
4450 ENTER @Agte506x;Buff$
4460 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:THRU ";Port(J);",";Port(I)
4470 OUTPUT @Agte506x;"*OPC?"
4480 ENTER @Agte506x;Buff$
4490 NEXT J
4500 NEXT I
4510 !
4520 ! Done
4530 !
4540 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COLL:SAVE"
4550 PRINT "Done"
4560 SUBEND
4570!
4580 Get_nop: SUB Get_nop(@Agte506x,INTEGER Nop,Ch$)
4590 ! Get All Segment's Points
4600 OUTPUT @Agte506x;":SENS"&Ch$&":SEGM:SWE:POIN?"
4610 ENTER @Agte506x;Nop
4620 SUBEND
4630 Exec_error_term: SUB Exec_error_term(@Agte506x,Rw$,Id$,Ch$,INTEGER
Idx,Nop,Respons,Stimulas,REAL Stok(*))
4640 INTEGER Ii
4650 REAL Error_term_data(1:5000)
4660 !
4670 DISP CHR$(138)&" Wait ..."&CHR$(136)
4680 !
4690 REDIM Error_term_data(1:Nop*2)
4700 !
4710 SELECT Rw$
4720 CASE "WRITE"
4730 FOR Ii=1 TO Nop
4740 Error_term_data(2*Ii-1)=Stok(Idx,2*Ii-1)
4750 Error_term_data(2*Ii)=Stok(Idx,2*Ii)
4760 NEXT Ii
4770 !
4780 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COEF
"&Id$&",";Respons;",";Stimulas;",";Error_term_data(*)
4790 !
```

```
4800 CASE "READ"
4810 FOR Ii=1 TO Nop
4820 Error_term_data(2*Ii-1)=-999
4830 Error_term_data(2*Ii)=-999
4840 NEXT Ii
4850 !
4860 OUTPUT @Agte506x;":SENS"&Ch$&":CORR:COEF? "&Id$&";";Respons;";";Stimulas
4870 ENTER @Agte506x;Error_term_data(*)
4880 !
4890 CALL Data_plot(Id$,Respons,Stimulas,Nop,Error_term_data(*))
4900 !
4910 FOR Ii=1 TO Nop
4920 Stok(Idx,2*Ii-1)=Error_term_data(2*Ii-1)
4930 Stok(Idx,2*Ii)=Error_term_data(2*Ii)
4940 NEXT Ii
4950 !
4960 END SELECT
4970 SUBEND
4980!
4990 Data_plot: SUB Data_plot(Error_term$,INTEGER Respons,Stimulas,Nop,REAL
Error_term_data(*))
5000 INTEGER Ii,Pen(1:2)
5010 REAL Y_minmax(1:2)
5020 DIM Wk$[20]
5030 !
5040 CLEAR SCREEN
5050 GINIT
5060 GCLEAR
5070 !
5080 Pen(1)=3
5090 Pen(2)=4
5100 !
5110 ! Get Min Value and Max Value from all data
5120 Y_minmax(1)=MIN(Error_term_data(*))
5130 Y_minmax(2)=MAX(Error_term_data(*))
5150 !
5160 IF (Y_minmax(1)=Y_minmax(2)) AND (Y_minmax(1)=0) THEN
```

```
5170 Y_minmax(1)=1
5180 Y_minmax(2)=-1
5190 ELSE
5200 IF (Y_minmax(1)=Y_minmax(2)) THEN
5210 Y_minmax(1)=Y_minmax(1)*.5
5220 Y_minmax(2)=Y_minmax(2)*1.5
5230 END IF
5240 END IF
5250 !
5260 VIEWPORT 25*RATIO,80*RATIO,40,90
5270 WINDOW 1,Nop,Y_minmax(1),Y_minmax(2)
5280 FRAME
5290 !
5300 VIEWPORT 80*RATIO,100*RATIO,40,90
5310 WINDOW 0,2,0,2
5320 PEN Pen(1)
5330 CSIZE 2.5
5340 LORG 2
5350 MOVE .2,1.5
5360 DRAW .4,1.5
5370 MOVE .5,1.5
5380 PEN 1
5390 LABEL ":Real Value"
5400 !
5410 PEN Pen(2)
5420 MOVE .2,1
5430 DRAW .4,1
5440 MOVE .5,1
5450 PEN 1
5460 LABEL ":Image Value"
5470 !
5480 VIEWPORT 25*RATIO,80*RATIO,90,100
5490 WINDOW 0,2,0,2
5500 CSIZE 3
5510 LORG 5
5520 MOVE 1,1.2
5530 LABEL "Error Term:"&Error_term$
```

```
5540 !
5550 MOVE 1,.5
5560 LABEL "Respons Port:"&VAL$(Respons)&" Stimulas Port:"&VAL$(Stimulas)
5570 !
5580 VIEWPORT 0,25*RATIO,40,90
5590 WINDOW 0,2,0,2
5600 CLIP -10,10,-10,10
5610 LORG 8
5620 CSIZE 3
5630 !
5640 MOVE 1.9,0
5650 LABEL VAL$(Y_minmax(1))
5660 MOVE 1.9,2
5670 LABEL VAL$(Y_minmax(2))
5680 !
5690 VIEWPORT 25*RATIO,80*RATIO,30,40
5700 WINDOW 0,2,0,2
5710 CLIP -10,10,-10,10
5720 LORG 5
5730 MOVE 0,1.5
5740 LABEL VAL$(1)
5750 MOVE 2,1.5
5760 LABEL VAL$(Nop)
5770 !
5780 VIEWPORT 25*RATIO,80*RATIO,40,90
5790 WINDOW 1,Nop,Y_minmax(1),Y_minmax(2)
5800 FOR Ii=2 TO Nop
5820 PEN Pen(1)
5830 MOVE Ii-1,Error_term_data(2*(Ii-1)-1)
5840 DRAW Ii,Error_term_data(2*Ii-1)
5860 !
5870 PEN Pen(2)
5880 MOVE Ii-1,Error_term_data(2*(Ii-1))
5890 DRAW Ii,Error_term_data(2*Ii)
5900 NEXT Ii
5910 !
5920 PEN 1
```

```
5930 BEEP
5940 INPUT "Cont:push [Enter] key",Wk$
5950 SUBEND
5960!
```

Waiting for Trigger (OPC?)

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to use the :TRIG:SING command to wait until the measurement cycle is completed.

NOTE This sample program correctly runs when the maximum number of channels/traces is set to 4 channels/4 traces.

The sample program uses the :TRIG:SING command to start a sweep (measurement) cycle, uses the *OPC command to wait until the measurement cycle is completed, then prints a message and exits.

See Waiting for the End of Measurement for this programming.

Sample Program in Excel VBA

```
Sub trg_sing_Click()
  Dim defrm As Long
  Dim vi As Long
  Dim ContMode(4) As String
  Dim Result As String * 10
  Dim i As Integer
  Const TimeOutTime = 100000 ' TimeOut time should be greater than the measurement time.
  '
  ' Assign a GPIB address to the I/O pass.
  Call viOpenDefaultRM(defrm)
  Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
  Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
  '
  ' Store the settings of continuous initiation mode for each channel
  ' (on for channels 1 and 2; off for channels 3 and 4)
  ' into the array variable ContMode().
  ContMode(1) = "ON"
  ContMode(2) = "ON"
  ContMode(3) = "OFF"
  ContMode(4) = "OFF"
```

```
    '
    ' Turn on or off continuous initiation mode for each channel
    ' depending on the value of ContMode(*).
    Call viVPrintf(vi, ":DISP:SPL D12_34" & vbLf, 0)
    Call viVPrintf(vi, ":SENS1:SWE:TIME:AUTO OFF" & vbLf, 0)
    Call viVPrintf(vi, ":SENS1:SWE:TIME 5" & vbLf, 0)
    Call viVPrintf(vi, ":SENS2:SWE:TIME:AUTO OFF" & vbLf, 0)
    Call viVPrintf(vi, ":SENS2:SWE:TIME 3" & vbLf, 0)
    Call viVPrintf(vi, ":SENS3:SWE:TIME:AUTO OFF" & vbLf, 0)
    Call viVPrintf(vi, ":SENS3:SWE:TIME 1" & vbLf, 0)
    Call viVPrintf(vi, ":SENS4:SWE:TIME:AUTO OFF" & vbLf, 0)
    Call viVPrintf(vi, ":SENS4:SWE:TIME 3" & vbLf, 0)
    For i = 1 To 4
       Call viVPrintf(vi, ":INIT" & CStr(i) & ":CONT " & ContMode(i) & vbLf, 0)
    Next i
    ' Set the trigger source to Bus Trigger.
    Call viVPrintf(vi, ":TRIG:SOUR BUS" & vbLf, 0)
    '
    ' Trigger the instrument to start a sweep cycle.
    Call viVPrintf(vi, ":TRIG:SING" & vbLf, 0)
    '
    ' Execute the *OPC? command and wait until the command
    ' returns 1 (i.e., the measurement cycle is completed).
    Call viVPrintf(vi, "*OPC?" & vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    '
    ' Display a measurement completion message.
    Stat = MsgBox("Measurement complete", vbOKOnly)
    Call viClose(vi)
    Call viClose(defrm)
End Sub
```

Sample Program in HT Basic (trg_sing.htb)

```
10 DIM Cont_mode$(1:4)[9],Buff$[9]
20 INTEGER I
30 !
40 ASSIGN @Agte506x TO 717
50 !
```

```
60 Cont_mode$(1)="ON"
70 Cont_mode$(2)="ON"
80 Cont_mode$(3)="OFF"
90 Cont_mode$(4)="OFF"
150 !
160 FOR I=1 TO 4
170 OUTPUT @Agte506x;":INIT"&VAL$(I)&":CONT "&Cont_mode$(I)
180 NEXT I
190 OUTPUT @Agte506x;":TRIG:SOUR BUS"
200 !
210 OUTPUT @Agte506x;":TRIG:SING"
220 OUTPUT @Agte506x;"*OPC?"
230 ENTER @Agte506x;Buff$
240 !
250 PRINT "Measurement complete"
260 END
```

Waiting for Trigger (SRQ)

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to use an SRQ to detect the end of measurement.

NOTE This sample program correctly runs when the maximum number of channels/traces is set to 4 channels/4 traces.

The sample program sets up the trigger system, configures the instrument to properly generate an SRQ, and then triggers the instrument. When the instrument has generated an SRQ that indicates the end of measurement, the program exits after printing a measurement completion message.

See Waiting for the End of Measurement for this programming.

Sample Program in Excel VBA

```
Sub srq_meas_Click()
  Dim defrm As Long
  Dim vi As Long
  Dim ContMode(9) As String
  Dim Result As String * 10
  Dim i As Integer, StbStatus As Integer
  Const TimeOutTime = 100000 ' TimeOut time should be greater than the measurement time.
  '
  ' Assign a GPIB address to the I/O pass.
  Call viOpenDefaultRM(defrm)
  Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
  Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
  '
  ' Store the settings of continuous initiation mode for eachchannel
  ' (on for channels 1 and 2; off for channels 3 and 4)
  ' into the array variable ContMode().
  ContMode(1) = "ON"
  ContMode(2) = "ON"
  ContMode(3) = "OFF"
```

```
ContMode(4) = "OFF"
'
' Turn on or off continuous initiation mode for each channel
' depending on the value of ContMode(*).
Call viVPrintf(vi, ":DISP:SPL D12_34" & vbLf, 0)
Call viVPrintf(vi, ":SENS1:SWE:TIME:AUTO OFF" & vbLf, 0)
Call viVPrintf(vi, ":SENS1:SWE:TIME 5" & vbLf, 0)
Call viVPrintf(vi, ":SENS2:SWE:TIME:AUTO OFF" & vbLf, 0)
Call viVPrintf(vi, ":SENS2:SWE:TIME 3" & vbLf, 0)
Call viVPrintf(vi, ":SENS3:SWE:TIME:AUTO OFF" & vbLf, 0)
Call viVPrintf(vi, ":SENS3:SWE:TIME 1" & vbLf, 0)
Call viVPrintf(vi, ":SENS4:SWE:TIME:AUTO OFF" & vbLf, 0)
Call viVPrintf(vi, ":SENS4:SWE:TIME 3" & vbLf, 0)
For i = 1 To 4
   Call viVPrintf(vi, ":INIT" & CStr(i) & ":CONT " & ContMode(i) & vbLf, 0)
Next i
'
' Set the trigger source to Bus Trigger.
Call viVPrintf(vi, ":TRIG:SOUR BUS" & vbLf, 0)
'
' .
Call viVPrintf(vi, ":STAT:OPER:PTR 0" & vbLf, 0) 'Set 0 at all bits of Position Transition Filter
Call viVPrintf(vi, ":STAT:OPER:NTR 16" & vbLf, 0) 'Set 1 at bit 4 of Negative Transition Filter
Call viVPrintf(vi, ":STAT:OPER:ENAB 16" & vbLf, 0) 'Set 1 at bit 4 of Operation status enable
Call viVPrintf(vi, "*SRE 128" & vbLf, 0) 'Set 1 at bit 7 of Service Request Enable Register
Call viVPrintf(vi, "*CLS" & vbLf, 0) ' Clear Register.
'
Call viVPrintf(vi, "*TRG" & vbLf, 0) 'Make a trigger
' Wait until Status Byte Register became 192
Do
   Call viReadSTB(vi, StbStatus) ' Read Status Byte Register
   Range("B5").Value = StbStatus
Loop Until StbStatus = 192
'
' Display a measurement completion message.
Stat = MsgBox("Measurement complete", vbOKOnly)
' Close IO
```

```
    Call viClose(vi)
    Call viClose(defrm)
End Sub
```

Sample Program in HT Basic (srq_meas.htb)

```
10 DIM Cont_mode$(1:4)[9],Buff$[9]
20 INTEGER I
30 !
40 ASSIGN @Agte506x TO 717
50 !
60 Cont_mode$(1)="ON"
70 Cont_mode$(2)="ON"
80 Cont_mode$(3)="OFF"
90 Cont_mode$(4)="OFF"
150 !
160 FOR I=1 TO 4
170 OUTPUT @Agte506x;":INIT"&VAL$(I)&":CONT "&Cont_mode$(I)
180 NEXT I
190 OUTPUT @Agte506x;":TRIG:SOUR BUS"
200 !
210 OUTPUT @Agte506x;":STAT:OPER:PTR 0"
220 OUTPUT @Agte506x;":STAT:OPER:NTR 16"
230 OUTPUT @Agte506x;":STAT:OPER:ENAB 16"
240 OUTPUT @Agte506x;"*SRE 128"
250 OUTPUT @Agte506x;"*CLS"
260 OUTPUT @Agte506x;"*OPC?"
270 ENTER @Agte506x;Buff$
280 !
290 ON INTR 7 GOTO Meas_end
300 ENABLE INTR 7;2
310 OUTPUT @Agte506x;"*TRG"
320 PRINT "Waiting..."
330 Meas_wait: GOTO Meas_wait
340 Meas_end: OFF INTR 7
350 PRINT "Measurement Complete"
360 END
```

Description

Line 40

Assigns a GPIB address to the I/O pass.

Lines 60 to 90

These lines store the settings of continuous initiation mode for each channel (on for channels 1 and 2; off for channels 3 and 4) into the array variable Cont_mode$(*).

Lines 160 to 180

These lines turn ON or OFF continuous initiation mode for each channel depending on the value of Cont_mode$(*).

Line 190

Sets the trigger source to "Bus Trigger".

Lines 210 to 220

These lines configure the instrument so that operation status event register's bit 4 is set to 1 only when the operation status condition register's bit 4 is changed from 1 to 0 (negative transition).

Lines 230 to 240

These lines enable the operation status event register's bit 4 and status byte register's bit 7.

Lines 250 to 270

These lines clear the status byte register and operation status event register.

Lines 290 to 300

These lines set the branch target for an SRQ interrupt to enable SRQ interruptions.

Lines 310 to 320

These lines trigger the instrument and wait until the measurement cycle finishes.

Line 350

Displays a measurement completion message.

Error Detection (SRQ)

- Overview
- Sample Program in Excel VBA using VISA-COM
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to use an SRQ to detect the occurrence of an error.

This program sets SRQs and then intentionally sends an invalid parameter to generate an error to be handled by this program. In the error handling part, this program examines the error, displays the error number and error message, and then displays the message indicating the suspension of the program. See Detecting Occurrence of an Error for this programming.

NOTE The sequence interception by an error can not be performed on Excel VBA using VISA.

Sample Program in Excel VBA using VISA-COM

```
Dim Age506x As VisaComLib.FormattedIO488
Option Explicit
Implements VisaComLib.IEventHandler

Private Sub IEventHandler_HandleEvent(ByVal vi As VisaComLib.IEventManager, ByVal SRQevent As
VisaComLib.IEvent, ByVal userHandle As Long)
  On Error Resume Next
  Dim readErr As Variant
  Dim i As Integer
  Age506x.WriteString "SYST:ERR?"
  readErr = Age506x.ReadList
  MsgBox "Error : " & readErr(0) & " , " & readErr(1), vbOKOnly, "Error occured."
  End

End Sub

Private Sub Err_Detect_Click()
  Dim gmgr As VisaComLib.ResourceManager
  Dim SRQ As VisaComLib.IEventManager

  Dim strdmy As String
```

```
 On Error GoTo CommandButton1_Error

 Set gmgr = New VisaComLib.ResourceManager

 Set Age506x = New VisaComLib.FormattedIO488
 Set Age506x.IO = gmgr.Open("GPIB0::17::INSTR")

 Set SRQ = Age506x.IO
 SRQ.InstallHandler EVENT_SERVICE_REQ, Me, 1, 0
 SRQ.EnableEvent EVENT_SERVICE_REQ, EVENT_HNDLR

 With Age506x
   .WriteString "*RST"
   .WriteString "*ESE 60"
   .WriteString "*SRE 32"
   .WriteString "*CLS"
   .WriteString "*OPC?"
   strdmy = .ReadString
 End With

 With Age506x
   .WriteString "CALC1:PAR:COUN 2"
   .WriteString "CALC1:PAR1:DEF S21"
   .WriteString "CALC1:PAR1:SEL"
   .WriteString "CALC1:FORM MLOG"

   .WriteString "CALC1:PAR2:DEF S11"
   .WriteString "CALC1:PAR2:SEL"
   .WriteString "CALC1:FORM LOG"
   .WriteString "*OPC?"
   strdmy = .ReadString
 End With

 SRQ.DisableEvent ALL_ENABLED_EVENTS, EVENT_ALL_MECH, 0

 Exit Sub
```

```
CommandButton1_Error:
  MsgBox "Error Occured. Error=" & vbTab & Err.Number & " , " & Err.Description
End Sub

Private Sub EndBtn_Click()
  End
End Sub
```

Sample Program in HT Basic (srq_err.bas)

```
10 DIM Buff$[9],Err_mes$[50]
20 INTEGER Err_no
30 !
40 ASSIGN @Agte506x TO 717
50 !
60 OUTPUT @Agte506x;"*ESE 60"
70 OUTPUT @Agte506x;"*SRE 32"
80 OUTPUT @Agte506x;"*CLS"
90 OUTPUT @Agte506x;"*OPC?"
100 ENTER @Agte506x;Buff$
110 !
120 ON INTR 7 GOTO Err_proc
130 ENABLE INTR 7;2
140 OUTPUT @Agte506x;":CALC1:PAR:COUN 2"
150 PRINT "Trace 1 Meas.Para: S21"
160 PRINT "Trace 1 Format : Log Mag"
170 OUTPUT @Agte506x;":CALC1:PAR1:DEF S21"
180 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
190 OUTPUT @Agte506x;":CALC1:FORM MLOG"
200 PRINT "Trace 2 Meas.Para: S11"
210 PRINT "Trace 2 Format : Log Mag"
220 OUTPUT @Agte506x;":CALC1:PAR2:DEF S11"
230 OUTPUT @Agte506x;":CALC1:PAR2:SEL"
240 OUTPUT @Agte506x;":CALC1:FORM LOG"
250 OUTPUT @Agte506x;"*OPC?"
260 ENTER @Agte506x;Buff$
270 GOTO Skip_err_proc
```

```
280 Err_proc: OFF INTR 7
290 OUTPUT @Agte506x;":SYST:ERR?"
300 ENTER @Agte506x;Err_no,Err_mes$
310 PRINT "Error occurred!!"
320 PRINT " No:";Err_no,"Description: "&Err_mes$
330 PRINT "PROGRAM INTERRUPT!!"
340 GOTO Prog_end
350 Skip_err_proc: PRINT "PROGRAM DONE."
360 Prog_end: END
```

Description

Line 40

Assigns a GPIB address to the I/O pass.

Lines 60 to 70

These lines enable bits 2, 3, 4 and 5 in the standard event status register and set bit 5 to 1 in the service request enable register.

Lines 80 to 100

These lines clear the status byte register, the standard event status register, and the error queue.

Lines 120 to 130

These lines set the branch target for an SRQ interrupt to enable SRQ interruptions.

Lines 140 to 260

These lines set the measurement parameters and their data formats for traces 1 and 2. An invalid parameter is given to the data format setting for trace 2, causing an error.

Lines 280 to 330

These lines define an error handler in the following way.

Lines 290 to 300: These lines retrieve the error number and error messages for the error from the error queue.

Lines 310 to 330 These lines display the message indicating the occurrence of the error, the error number, the error message, and the message showing that the program is suspended.

Line 350

Displays a closing message. Note that this message will not display unless this program is re-executed after setting a corrected parameter to the data format setting for trace 2.

Reading Data in ASCII Format

- Overview
- Sample Program in Excel VBA using VISA
- Sample Program in Excel VBA using VISA-COM
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to retrieve formatted data arrays in the ASCII transfer format.

This program holds the sweep on channel 1, then retrieves and displays the stimulus array for channel 1 and the formatted data array for trace 1.

See Retrieving Measurement Results for this programming.

Sample Program in Excel VBA using VISA

```
Sub read_asc_Click()
    '
    Dim defrm As Long
    Dim vi As Long
    Dim Result As String * 10000
    Dim Res As Variant
    Dim Res2 As Variant
    Dim Nop As Long
    Const TimeOutTime = 10000
    '
    ' Open the Analyzer
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    '
    ' Select Parameter 1
    Call viVPrintf(vi, ":CALC1:PAR1:SEL" + vbLf, 0)
    Call viVPrintf(vi, ":INIT1:CONT OFF" + vbLf, 0)
    Call viVPrintf(vi, ":ABOR" + vbLf, 0)
    '
    ' Read out NOP Data in ASCII transfer format
    Call viVPrintf(vi, ":SENS1:SWE:POIN?" + vbLf, 0)
    Call viVScanf(vi, "%t", Result)
```

```
    Nop = Val(Result)
    ReDim FMTData(Nop, 2)
    ReDim Freq(Nop)
    '
    ' Read out Measurement Data in ASCII transfer format
    Call viVPrintf(vi, ":FORM:DATA ASC" + vbLf, 0)
    Result = ""
    Call viVPrintf(vi, ":CALC1:DATA:FDAT?" + vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    Res = Split(Result, ",")
    '
    Range("A6:D1607").Clear 'Clear cells of Excel
    '
    ' Write data in cells of Excel
    j = 0
    For i = 1 To Nop
        Cells(i + 5, 1) = i
        Cells(i + 5, 3) = Val(Res(j))
        Cells(i + 5, 4) = Val(Res(j + 1))
        j = j + 2
    Next i
    '
    ' Read out Measurement Frequency Data in ASCII transfer format
    Result = ""
    Call viVPrintf(vi, ":SENS1:FREQ:DATA?" + vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    Res2 = Split(Result, ",")
    '
    '  Write data in cells of Excel
    For i = 1 To Nop
        Cells(i + 5, 2) = Val(Res2(i - 1))
    Next i
    '
    ' Close the Analyzer
    Call viClose(vi)
    Call viClose(defrm)
End Sub
```

Sample Program in Excel VBA using VISA-COM

```
Private Sub Read_ASC_Click()
  Dim ReadData() As Double
  Dim Poin As Integer
  Dim FreqData() As Double

  '*** The variables of the resource manager and the instrument I/O are declared.
  Dim ioMgr As VisaComLib.ResourceManager
  Dim Age506x As VisaComLib.FormattedIO488

  '*** The memory area of the resource manager and the instrument I/O are acquired.
  Set ioMgr = New VisaComLib.ResourceManager
  Set Age506x = New VisaComLib.FormattedIO488

  '*** Open the instrument.
  Set Age506x.IO = ioMgr.Open("GPIB0::17::INSTR")
  Age506x.IO.timeout = 10000

  '*** Abort sweeping of channel1/trace1.
  Age506x.WriteString ":CALC1:PAR1:SEL", True
  Age506x.WriteString ":INIT1:CONT OFF", True
  Age506x.WriteString ":ABOR", True

  '*** Set the ascii format.
  Age506x.WriteString ":FORM:DATA ASC", True

  '*** Get num of point and the stimulus data.
  Age506x.WriteString ":SENS1:SWE:POIN?", True
  Poin = Age506x.ReadNumber
  ReDim FreqData(Poin - 1)
  Age506x.WriteString ":SENS1:FREQ:DATA?", True
  FreqData() = Age506x.ReadList(ASCIIType_R8, ",")

  '*** Get the measurement data.
  ReDim ReadData(Poin * 2 - 1)
  Age506x.WriteString ":CALC1:DATA:FDAT?", True
  ReadData() = Age506x.ReadList(ASCIIType_R8, ",")
```

```
  '*** set data for new sheet
  Sheets.Add
  ActiveSheet.Name = Format(Year(Date), "0000") & Format(Month(Date), "00") & Format(Day(Date),
"00") _
                & Format(Hour(Time), "00") & Format(Minute(Time), "00") & Format(Second(Time), "00")
  ActiveSheet.Cells(5, 3) = "Frequency"
  ActiveSheet.Cells(5, 4) = "Data 1"
  ActiveSheet.Cells(5, 5) = "Data 2"
  For i = 1 To Poin
    ActiveSheet.Cells(i + 5, 3) = FreqData(i - 1)

    ActiveSheet.Cells(i + 5, 4).Value = ReadData(i * 2 - 2)
    ActiveSheet.Cells(i + 5, 5).Value = ReadData(i * 2 - 1)
  Next i

  '*** end procedure
  Age506x.IO.Close
End Sub
```

Sample Program in HT Basic (read_asc.htb)

```
10 REAL Fdata(1:1601,1:2),Freq(1:1601)
20 DIM Img$[30]
30 INTEGER Nop,I
40 !
50 ASSIGN @Agte506x TO 717
60 !
70 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
80 OUTPUT @Agte506x;":INIT1:CONT OFF"
90 OUTPUT @Agte506x;":ABOR"
100 OUTPUT @Agte506x;":SENS1:SWE:POIN?"
110 ENTER @Agte506x;Nop
120 REDIM Fdata(1:Nop,1:2),Freq(1:Nop)
130 !
140 ! Reading out in ASCII transfer format
150 !
160 OUTPUT @Agte506x;":FORM:DATA ASC"
170 !
```

```
180 OUTPUT @Agte506x;":CALC1:DATA:FDAT?"
190 ENTER @Agte506x;Fdata(*)
200 OUTPUT @Agte506x;":SENS1:FREQ:DATA?"
210 ENTER @Agte506x;Freq(*)
220 !
230 ! Displaying
240 !
250 OUTPUT @Agte506x;":CALC1:FORM?"
260 ENTER @Agte506x;Fmt$
270 SELECT Fmt$
280 CASE "MLOG","PHAS","GDEL","MLIN","SWR","REAL","IMAG","UPH"
290 Img$="MD.4DE,2X,MD.6DE"
300 PRINT " Frequency Data"
310 FOR I=1 TO Nop
320 PRINT USING Img$;Freq(I),Fdata(I,1)
330 NEXT I
340 CASE ELSE
350 Img$="MD.4DE,2X,MD.6DE,2X,MD.6DE"
360 PRINT " Frequency Data1 Data2"
370 FOR I=1 TO Nop
380 PRINT USING Img$;Freq(I),Fdata(I,1),Fdata(I,2)
390 NEXT I
400 END SELECT
410 !
420 END
```

Reading Data in Binary Format

- Overview
- Sample Program in Excel VBA using VISA
- Sample Program in Excel VBA using VISA-COM
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to retrieve formatted data arrays in the Binary transfer format.

This program holds the sweep on channel 1, then retrieves and displays the stimulus array.

See Retrieving Measurement Results for this programming.

Sample Program in Excel VBA using VISA

```
Sub read_bin_Click()
    Dim defrm As Long
    Dim vi As Long
    Dim Result As String * 10000
    Dim Res() As Double
    Dim Nop As Long
    Const TimeOutTime = 10000
    ' Open Analyzer
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    '
    Call viVPrintf(vi, ":CALC1:PAR1:SEL" + vbLf, 0)
    Call viVPrintf(vi, ":INIT1:CONT OFF" + vbLf, 0)
    Call viVPrintf(vi, ":ABOR" + vbLf, 0)
    '
    ' Reading out Measurement Frequency Data in Binary transfer format
    Call viVPrintf(vi, ":FORM:DATA REAL" + vbLf, 0)
    Call viVPrintf(vi, ":CALC1:DATA:FDAT?" + vbLf, 0)
    Call Scpi_read_binary_double_array(vi, Res, Nop)
    '
    ' Write data in cells of Excel
    Range("A6:D1607").Clear
```

```
    For i = 0 To Nop - 1
        j = i Mod 2
        k = i \ 2
        Cells(k + 6, j + 3).Value = Res(i)
    Next i
    '
    ' Read out Measurement Frequency Data in Binary transfer format
    Call viVPrintf(vi, ":SENS1:FREQ:DATA?" + vbLf, 0)
    Call Scpi_read_binary_double_array(vi, Res, Nop)
    '
    ' Write data in cells of Excel
    For i = 0 To Nop - 1
        Cells(i + 6, 1) = i + 1
        Cells(i + 6, 2).Value = Res(i)
    Next i
    ' Close
    Call viClose(vi)
    Call viClose(defrm)
End Sub
    '==================================
    ' BinaryAry Read Subroutine
    '==================================
Sub Scpi_read_binary_double_array(vi As Long, data() As Double, Nop As Long)
    Dim dblArray(10000) As Double
    Dim paramsArray(3) As Long
    Dim err As Long
    Dim i As Long
    Dim lf_eoi As String * 1

    Nop = UBound(dblArray) - LBound(dblArray) + 1
    paramsArray(0) = VarPtr(Nop)
    paramsArray(1) = VarPtr(dblArray(0))
    err = viVScanf(vi, "%#Zb%1t", paramsArray(0))
    If err <> 0 Then MsgBox "Binary Error"

    ReDim data(Nop - 1)
    For i = 0 To Nop - 1
```

```
        data(i) = dblArray(i)
    Next
End Sub
```

Sample Program in Excel VBA using VISA-COM

```
Private Sub Read_BINARY_Click()
  Dim ReadData() As Double
  Dim Poin As Integer
  Dim FreqData() As Double
  '*** The variables of the resource manager and the instrument I/O are declared.
  Dim ioMgr As VisaComLib.ResourceManager
  Dim Age506x As VisaComLib.FormattedIO488
  '*** The memory area of the resource manager and the instrument I/O are acquired.
  Set ioMgr = New VisaComLib.ResourceManager
  Set Age506x = New VisaComLib.FormattedIO488

  '*** Open the instrument.
  Set Age506x.IO = ioMgr.Open("GPIB0::17::INSTR")
  Age506x.IO.Timeout = 10000

  '*** Abort sweeping of channel1/trace1.
  Age506x.WriteString ":CALC1:PAR1:SEL", True
  Age506x.WriteString ":INIT1:CONT OFF", True
  Age506x.WriteString ":ABOR", True
  '*** Set the binary format.
  Age506x.WriteString ":FORM:DATA REAL", True

  '*** Get num of point and the stimulus data.
  Age506x.WriteString ":SENS1:SWE:POIN?", True
  Poin = Age506x.ReadNumber
  Age506x.WriteString ":SENS1:FREQ:DATA?", True
  FreqData = Age506x.ReadIEEEBlock(BinaryType_R8, False, True)
  '*** Get the measurement data.
  Age506x.WriteString ":CALC1:DATA:FDAT?", True
  ReadData = Age506x.ReadIEEEBlock(BinaryType_R8, False, True)

  '*** set data for new sheet
  Sheets.Add
```

```
  ActiveSheet.Name = Format(Year(Date), "0000") & Format(Month(Date), "00") & Format(Day(Date),
"00") _
                & Format(Hour(Time), "00") & Format(Minute(Time), "00") & Format(Second(Time), "00")
  ActiveSheet.Cells(5, 3) = "Frequency"
  ActiveSheet.Cells(5, 4) = "Data 1"
  ActiveSheet.Cells(5, 5) = "Data 2"
  For i = 1 To Poin
     ActiveSheet.Cells(i + 5, 3) = FreqData(i - 1)

     ActiveSheet.Cells(i + 5, 4).Value = ReadData(i * 2 - 2)
     ActiveSheet.Cells(i + 5, 5).Value = ReadData(i * 2 - 1)
  Next i

  '*** end procedure
  Age506x.WriteString ":FORM:DATA ASC", True
  Age506x.IO.Close
End Sub
```

Sample Program in HT Basic (read_bin.htb)

```
10 REAL Fdata(1:1601,1:2),Freq(1:1601)
20 DIM Buff$[9],Img$[30]
30 INTEGER Nop,I
40 !
50 ASSIGN @Agte506x TO 717
60 ASSIGN @Binary TO 717;FORMAT OFF
70 !
80 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
90 OUTPUT @Agte506x;":INIT1:CONT OFF"
100 OUTPUT @Agte506x;":ABOR"
110 OUTPUT @Agte506x;":SENS1:SWE:POIN?"
120 ENTER @Agte506x;Nop
130 REDIM Fdata(1:Nop,1:2),Freq(1:Nop)
140 !
150 ! Reading out in binary transfer format
160 !
170 OUTPUT @Agte506x;":FORM:DATA REAL"
180 !
190 OUTPUT @Agte506x;":CALC1:DATA:FDAT?"
```

```
200 ENTER @Agte506x USING "#,8A";Buff$
210 ENTER @Binary;Fdata(*)
220 ENTER @Agte506x USING "#,1A";Buff$
230 OUTPUT @Agte506x;":SENS1:FREQ:DATA?"
240 ENTER @Agte506x USING "#,8A";Buff$
250 ENTER @Binary;Freq(*)
260 ENTER @Agte506x USING "#,1A";Buff$
270 !
280 ! Displaying
290 !
300 OUTPUT @Agte506x;":CALC1:FORM?"
310 ENTER @Agte506x;Fmt$
320 SELECT Fmt$
330 CASE "MLOG","PHAS","GDEL","MLIN","SWR","REAL","IMAG","UPH"
340 Img$="MD.4DE,2X,MD.6DE"
350 PRINT " Frequency Data"
360 FOR I=1 TO Nop
370 PRINT USING Img$;Freq(I),Fdata(I,1)
380 NEXT I
390 CASE ELSE
400 Img$="MD.4DE,2X,MD.6DE,2X,MD.6DE"
410 PRINT " Frequency Data1 Data2"
420 FOR I=1 TO Nop
430 PRINT USING Img$;Freq(I),Fdata(I,1),Fdata(I,2)
440 NEXT I
450 END SELECT
460 !
470 END
```

Writing Data in Ascii Format

- Overview
- Sample Program in Excel VBA using VISA
- Sample Program in Excel VBA using VISA-COM
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

The sample program demonstrates to Write Formatted Data Arrays in Using the ASCII Transfer Format

See Entering Data into a Trace for this programming.

NOTE The sample programs of Excel VBA (using VISA/VISA-COM) does not create the Ch2, and the measurement point is not set. It is necessary to create Ch2 and to set the measurement point to a prescribed measurement point before the program is executed.

Sample Program in Excel VBA using VISA

```
Private Sub Write_ASC_Click()
  '*** Before this program is executed, create the channel 2,
  '*** and match the number of points and data format
  '*** of channel 2 to the transmitted data.
  Dim defrm As Long
  Dim Age506x As Long
  Dim Poin As Integer
  Dim Buf As String * 16
  Dim WriteData() As Double
  Dim AryPtr As Long
  Dim WrtFmt As String

  Call viOpenDefaultRM(defrm)
  '*** Open the instrument, and set times 10sec for timeout.
  Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, Age506x)
  Call viSetAttribute(Age506x, VI_ATTR_TMO_VALUE, 10000)

  '*** Abort sweeping of channel2/trace1.
  Call viVPrintf(Age506x, ":CALC2:PAR1:SEL" + vbLf, 0)
  Call viVPrintf(Age506x, ":INIT2:CONT OFF" + vbLf, 0)
```

```
  Call viVPrintf(Age506x, ":ABOR" + vbLf, 0)
  '*** Set the ascii format.
  Call viVPrintf(Age506x, ":FORM:DATA ASC" + vbLf, 0)

  '*** Get num of point.
  Call viVPrintf(Age506x, ":SENS1:SWE:POIN?" + vbLf, 0)
  Call viVScanf(Age506x, "%t", Buf)
  Poin = CInt(Buf)
  ReDim WriteData(Poin * 2 - 1)

  '*** Set data for array variable, and send data for E506x.
  For i = 1 To Poin
    WriteData(i * 2 - 2) = ActiveSheet.Cells(i + 5, 4).Value
    WriteData(i * 2 - 1) = ActiveSheet.Cells(i + 5, 5).Value
  Next i
  AryPtr = VarPtr(WriteData(0))
  Call viVPrintf(Age506x, ":CALC2:DATA:FDAT ", 0)

  WrtFmt = "%," & Poin * 2 & "lf"
  Call viVPrintf(Age506x, WrtFmt, AryPtr)
  Call viVPrintf(Age506x, vbLf, 0)

  '*** end procedure
  Call viClose(Age506x)
  Call viClose(defrm)
End Sub
```

Sample Program in Excel VBA using VISA-COM

```
Private Sub Write_ASC_Click()
  '*** Before this program executes, create the channel 2,
  '*** and match the number of points and data format
  '*** of channel 2 to the transmitted data.
  Dim WriteData() As Double
  Dim Poin As Integer
  '*** The variables of the resource manager and the instrument I/O are declared.
  Dim ioMgr As VisaComLib.ResourceManager
  Dim Age506x As VisaComLib.FormattedIO488
```

```
'*** The memory area of the resource manager and the instrument I/O are acquired.
Set ioMgr = New VisaComLib.ResourceManager
Set Age506x = New VisaComLib.FormattedIO488

'*** Open the instrument.
Set Age506x.IO = ioMgr.Open("GPIB0::17::INSTR")
Age506x.IO.Timeout = 10000

'*** Abort sweeping of channel2/trace1.
Age506x.WriteString ":CALC2:PAR1:SEL", True
Age506x.WriteString ":INIT2:CONT OFF", True
Age506x.WriteString ":ABOR", True
'*** Set the ascii format.
Age506x.WriteString ":FORM:DATA ASC", True

'*** Get num of point.
Age506x.WriteString ":SENS1:SWE:POIN?", True
Poin = Age506x.ReadNumber

ReDim WriteData(Poin * 2 - 1) As Double
'*** Set data for array variable, and send data for E506x.
For i = 1 To Poin
  WriteData(i * 2 - 2) = ActiveSheet.Cells(i + 5, 4).Value
  WriteData(i * 2 - 1) = ActiveSheet.Cells(i + 5, 5).Value
Next i
Age506x.WriteString "CALC2:DATA:FDAT ", False
Age506x.WriteList WriteData, ASCIIType_R8, ",", True

'*** end procedure
Age506x.IO.Close
End Sub
```

Sample Program in HT Basic (write_a.htb)

```
10 REAL Freq,Fdata(1:1601,1:2)
20 DIM File$[300]
30 INTEGER Nop
40 !
```

```
50 ASSIGN @Agte506x TO 717
60 !
70 CALL Inp_file_name(File$)
80 !
90 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
100 OUTPUT @Agte506x;":INIT1:CONT OFF"
110 OUTPUT @Agte506x;":ABOR"
120 !
130 OUTPUT @Agte506x;":SENS1:SWE:POIN?"
140 ENTER @Agte506x;Nop
150 REDIM Fdata(1:Nop,1:2)
160 !
170 ON ERROR GOTO File_error
180 ASSIGN @File TO File$
190 ENTER @File USING "K";Buff$
200 ENTER @File USING "K";Buff$
210 ENTER @File USING "K";Buff$
220 FOR I=1 TO Nop
230 ENTER @File USING "19D,2X,19D,2X,19D";Freq,Fdata(I,1),Fdata (I,2)
240 NEXT I
250 ASSIGN @File TO *
260 OFF ERROR
270 !
280 OUTPUT @Agte506x;":FORM:DATA ASC"
290 !
300 OUTPUT @Agte506x;":CALC1:DATA:FDAT ";Fdata(*)
310 !
320 GOTO Prog_end
330 !
340 File_error: OFF ERROR
350 PRINT "############ ERROR ############"
360 PRINT File$&" is NOT exist."
370 PRINT " or"
380 PRINT File$&" has UNSUITABLE data."
390 !
400 Prog_end: END
410 !=========================================
```

```
420 ! File Name Input Function
430 !=========================================
440 SUB Inp_file_name(Inp_name$)
450 DIM Inp_char$[9]
460 ON ERROR GOTO Inp_start
470 Inp_start: !
480 PRINT "Input File Name!"
490 INPUT "Name?",Inp_name$
500 PRINT "Input Name: "&Inp_name$
510 INPUT "OK? [Y/N]",Inp_char$
520 IF UPC$(Inp_char$)<>"Y" THEN Inp_start
530 OFF ERROR
540 SUBEND
```

Description

Line 50

Assigns a GPIB address to the I/O pass.

Line 70

Passes control to a subprogram named Inp_file_name, which lets the user input a file name, and then stores the returned file name into the File$ variable. For more information on the Inp_file_name subprogram, refer to the description in Using the Binary Transfer Format to write Formatted Data Arrays.

Lines 90 to 110

These lines set channel 1's active trace to trace 1 and hold the sweep.

Lines 130 to 140

These lines retrieve the number of points in channel 1 and stores that number into the Nop variable.

Line 150

Resizes the Fdata array based on the value of the Nop variable (the number of points).

Line 170

This line points to the statement block to be executed if an error occurs in retrieving data from the file (for example, if no file matches File$).

Lines 180 to 260

These lines retrieve the formatted data from the file identified by File$, and store the data into the Fdata array.

Line 280

Sets the data transfer format to ASCII.

Line 300

Writes Fdata into the formatted data array for the active trace (trace 1) in channel 1.

Lines 340 to 380

This statement block is executed if an error occurs in retrieving data from the file.

Writing Data in Binary Format

- Overview
- Sample Program in Excel VBA using VISA
- Sample Program in Excel VBA using VISA-COM
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

The sample program demonstrates to Write Formatted Data Arrays in Using the Binary Transfer Format.

See Entering Data into a Trace for this programming.

NOTE The sample programs of Excel VBA (using VISA/VISA-COM) does not create the Ch2, and the measurement point is not set. It is necessary to create Ch2 and to set the measurement point to a prescribed measurement point before the program is executed.

Sample Program in Excel VBA using VISA

```
Private Sub Write_BINARY_Click()
  '*** Before this program is executed, create the channel 2,
  '*** and match the number of points and data format
  '*** of channel 2 to the transmitted data.
  Dim defrm As Long
  Dim Age506x As Long
  Dim Poin As Integer
  Dim Buf As String * 16
  Dim WriteData() As Double
  Dim WrtPtr As Long
  Dim WrtFmt As String

  Call viOpenDefaultRM(defrm)
  '*** Open the instrument.
  Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, Age506x)
  Call viSetAttribute(Age506x, VI_ATTR_TMO_VALUE, 10000)

  '*** Abort sweeping of channel2/trace1.
  Call viVPrintf(Age506x, ":CALC2:PAR1:SEL" + vbLf, 0)
  Call viVPrintf(Age506x, ":INIT2:CONT OFF" + vbLf, 0)
```

```
  Call viVPrintf(Age506x, ":ABOR" + vbLf, 0)
  '*** Set the binary format.
  Call viVPrintf(Age506x, ":FORM:DATA REAL" + vbLf, 0)
  '*** Get num of point, and re-define array size of data.
  Call viVPrintf(Age506x, ":SENS1:SWE:POIN?" + vbLf, 0)
  Call viVScanf(Age506x, "%t", Buf)
  Poin = CInt(Buf)
  ReDim WriteData(Poin * 2 - 1)
  WrtPtr = VarPtr(WriteData(0))

  '*** Set data for array variable.
  For i = 1 To Poin
    WriteData(i * 2 - 2) = ActiveSheet.Cells(i + 5, 4).Value
    WriteData(i * 2 - 1) = ActiveSheet.Cells(i + 5, 5).Value
  Next i
  '*** set Format code & Modifier.
  WrtFmt = "%#" & Trim(CStr(Poin * 2)) & "Zb"
  '*** send data for E506x.
  Call viVPrintf(Age506x, ":CALC2:DATA:FDAT ", 0)
  Call viVPrintf(Age506x, WrtFmt + vbLf, WrtPtr)

  '*** end procedure
  Call viVPrintf(Age506x, ":FORM:DATA ASC" + vbLf, 0)
  Call viClose(Age506x)
  Call viClose(defrm)

End Sub
```

Sample Program in Excel VBA using VISA-COM

```
Private Sub Write_BINARY_Click()
  '*** Before this program executes, create the channel 2,
  '*** and match the number of points and data format
  '*** of channel 2 to the transmitted data.
  Dim WriteData() As Double
  Dim Poin As Integer
  '*** The variables of the resource manager and the instrument I/O are declared.
  Dim ioMgr As VisaComLib.ResourceManager
```

```
  Dim Age506x As VisaComLib.FormattedIO488
  '*** The memory area of the resource manager and the instrument I/O are acquired.
  Set ioMgr = New VisaComLib.ResourceManager
  Set Age506x = New VisaComLib.FormattedIO488

  '*** Open the instrument.
  Set Age506x.IO = ioMgr.Open("GPIB0::17::INSTR")
  Age506x.IO.Timeout = 10000

  '*** Abort sweeping of channel2/trace1.
  Age506x.WriteString ":CALC2:PAR1:SEL", True
  Age506x.WriteString ":INIT2:CONT OFF", True
  Age506x.WriteString ":ABOR", True
  '*** Set the binary format.
  Age506x.WriteString ":FORM:DATA REAL", True

  '*** Get num of point.
  Age506x.WriteString ":SENS1:SWE:POIN?", True
  Poin = Age506x.ReadNumber
  ReDim WriteData(Poin * 2 - 1) As Double
  '*** Set data for array variable, and send data for E506x.
  For i = 1 To Poin
    WriteData(i * 2 - 2) = ActiveSheet.Cells(i + 5, 4).Value
    WriteData(i * 2 - 1) = ActiveSheet.Cells(i + 5, 5).Value
  Next i
  Age506x.WriteIEEEBlock ":CALC2:DATA:FDAT ", WriteData, True

  '*** end procedure
  Age506x.WriteString ":FORM:DATA ASC", True
  Age506x.IO.Close

End Sub
```

Sample Program in HT Basic (write_b.htb)

```
10 REAL Freq,Fdata(1:1601,1:2)
20 DIM File$[300],Header$[10]
30 INTEGER Nop
```

```
40 !
50 ASSIGN @Agte506x TO 717
60 ASSIGN @Binary TO 717;FORMAT OFF
70 CALL Inp_file_name(File$)
80 !
90 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
100 OUTPUT @Agte506x;":INIT1:CONT OFF"
110 OUTPUT @Agte506x;":ABOR"
120 !
130 OUTPUT @Agte506x;":SENS1:SWE:POIN?"
140 ENTER @Agte506x;Nop
150 REDIM Fdata(1:Nop,1:2)
160 !
170 ON ERROR GOTO File_error
180 ASSIGN @File TO File$
190 ENTER @File USING "K";Buff$
200 ENTER @File USING "K";Buff$
210 ENTER @File USING "K";Buff$
220 FOR I=1 TO Nop
230 ENTER @File USING "19D,2X,19D,2X,19D";Freq,Fdata(I,1),Fdata (I,2)
240 NEXT I
250 ASSIGN @File TO *
260 OFF ERROR
270 !
280 OUTPUT @Agte506x;":FORM:DATA REAL"
290 Header$="#6"&IVAL$(8*2*Nop,10)
300 OUTPUT @Agte506x;":CALC1:DATA:FDAT ";Header$;
310 OUTPUT @Binary;Fdata(*),END
320 GOTO Prog_end
330 !
340 File_error: OFF ERROR
350 PRINT "############ ERROR ############"
360 PRINT File$&" is NOT exist."
370 PRINT " or"
380 PRINT File$&" has UNSUITABLE data."
390 !
400 Prog_end: END
```

```
410 !=========================================
420 ! File Name Input Function
430 !=========================================
440 SUB Inp_file_name(Inp_name$)
450 DIM Inp_char$[9]
460 ON ERROR GOTO Inp_start
470 Inp_start: !
480 PRINT "Input File Name!"
490 INPUT "Name?",Inp_name$
500 PRINT "Input Name: "&Inp_name$
510 INPUT "OK? [Y/N]",Inp_char$
520 IF UPC$(Inp_char$)<>"Y" THEN Inp_start
530 OFF ERROR
540 SUBEND
```

Description

Lines 50 to 60

Assigns a GPIB address to the I/O pass.

Line 70

Passes control to a subprogram named Inp_file_name, which lets the user input a file name, and then stores the returned file name into the File$ variable.

Lines 90 to 110

These lines set channel 1's active trace to trace 1 and hold the sweep.

Lines 130 to 140

These lines retrieve the number of points in channel 1 and stores that number into the Nop variable.

Line 150

Resizes the Fdata array based on the value of the Nop variable (the number of points).

Line 170

This line points to the statement block to be executed if an error occurs in retrieving data from the file (for example, if no file matches File$).

Lines 180 to 260

These lines retrieve the formatted data from the file identified by File$, and store the data into the Fdata array.

Line 280

Sets the data transfer format to binary.

Line 290

Creates the data header and stores it into the Header$ variable.

Line 300

Sends the command that writes data into the formatted data array for the active trace (trace 1) in channel 1, following it with the data header (Header$).

Line 310

Sends the data itself (Fdata), following it with a message terminator.

NOTE Because binary data must be written without being formatted, the program uses an I/O path (@Binary) that is configured to support writing unformatted data.

Lines 340 to 380

This statement block is executed if an error occurs in retrieving data from the file.

The Inp_file_name subprogram in lines 440 to 540, which is used to enter a save filename, is described below.

Line 460

Allows the user to return to the entry start line and re-enter the data if an error (such as an invalid entry) occurs while entering the target file name.

Lines 480 to 490

These lines prompt the user to enter the target file name. The program does not continue till the user actually enters the file name.

Lines 500 to 510

These lines display the entered file name and waits for a confirmation entry (y/n key).

Line 520

Returns to the entry start line if the key the user pressed in line 510 is not the y key.

Peak Search

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to search for peaks using the Marker Search feature and analysis commands.

This program works in two steps: first, it uses Marker Search to search for the maximum positive peak and displays the results; second, it uses analysis commands to search for all positive peaks and displays the results.

See Searching for Positions That Match Specified Criteria for this programming.

Sample Program in Excel VBA

```
Private Sub PeakSearch_Click()
    Dim defrm As Long
    Dim vi As Long
    Const TimeOutTime = 20000
    '
    Dim Buff As String, Img As String, Err_msg As String
    Dim Excursion As String, Freq As String * 20, Resp As Variant, PeakPoint As Variant
    Dim Poin As String * 5, Result As String * 1000, errmsg As String * 20

    Excursion = "0.5"
    ' Open Analyzer
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
    Call viVPrintf(vi, "*CLS" & vbLf, 0)
    '
    ' Setup Analyzer
    Call viVPrintf(vi, ":SENS1:FREQ:CENT 947.5E6" & vbLf, 0)
    Call viVPrintf(vi, ":SENS1:FREQ:SPAN 200E6" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:PAR1:DEF S11" & vbLf, 0)
    Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)
    '
```

```
Call viVPrintf(vi, ":CALC1:PAR1:SEL" & vbLf, 0)
Call viVPrintf(vi, ":CALC1:MARK1:FUNC:TYPE PEAK" & vbLf, 0)
Call viVPrintf(vi, ":CALC1:MARK1:FUNC:PEXC " & Excursion & vbLf, 0)
Call viVPrintf(vi, ":CALC1:MARK1:FUNC:PPOL POS" & vbLf, 0)
Call viVPrintf(vi, ":CALC1:MARK1:FUNC:EXEC" & vbLf, 0)
Call ErrorCheck(vi)
Call viVPrintf(vi, ":CALC1:MARK1:X?" & vbLf, 0)
Call viVScanf(vi, "%t", Freq)
'
Call viVPrintf(vi, ":CALC1:MARK1:Y?" & vbLf, 0)
Call viVScanf(vi, "%t", Result)
'
Resp = Split(Result, ",")
Cells(5, 5).Value = Val(Freq)
Cells(5, 6).Value = Val(Resp(0))
'
Call viVPrintf(vi, ":CALC1:FUNC:DOM OFF" & vbLf, 0)
Call viVPrintf(vi, ":CALC1:FUNC:TYPE APE" & vbLf, 0)
Call viVPrintf(vi, ":CALC1:FUNC:PEXC " & Excursion & vbLf, 0)
Call viVPrintf(vi, ":CALC1:FUNC:PPOL NEG" & vbLf, 0)
Call viVPrintf(vi, ":CALC1:FUNC:EXEC" & vbLf, 0)
Call ErrorCheck(vi)
'
Call viVPrintf(vi, ":CALC1:FUNC:POIN?" & vbLf, 0)
Call viVScanf(vi, "%t", Poin)
Call viVPrintf(vi, ":CALC1:FUNC:DATA?" & vbLf, 0)
Call viVScanf(vi, "%t", Result)
PeakPoint = Split(Result, ",")
'
j = 0
For i = 1 To Val(Poin)
   Cells(6 + i, 5).Value = Val(PeakPoint(j))
   Cells(6 + i, 6).Value = Val(PeakPoint(j + 1))
   j = j + 2
Next i
'
Call viClose(vi)
```

```
    Call viClose(defrm)
    '
End Sub
Sub ErrorCheck(vi)
    Dim err As String * 50, ErrNo As Variant, Response As VbMsgBoxResult
    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)
    ErrNo = Split(err, ",")
    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)
    End If
End Sub
```

Sample Program in HT Basic (search.htb)

```
10 DIM Buff$[9],Img$[50],Err_msg$[100]
20 REAL Excursion,Freq,Resp,Result(1:100,1:2)
30 INTEGER Poin,Err_no
40 !
50 ASSIGN @Agte506x TO 717
60 Excursion=.5
70 !
80 OUTPUT @Agte506x;"*ESE 60"
90 OUTPUT @Agte506x;"*SRE 32"
100 OUTPUT @Agte506x;"*CLS"
110 OUTPUT @Agte506x;"*OPC?"
120 ENTER @Agte506x;Buff$
130 ON INTR 7 GOTO Err
140 ENABLE INTR 7;2
150 !
160 PRINT "Maximum Peak Search using Marker 1"
170 !
180 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
190 OUTPUT @Agte506x;":CALC1:MARK1:FUNC:TYPE PEAK"
200 OUTPUT @Agte506x;":CALC1:MARK1:FUNC:PEXC ";Excursion
210 OUTPUT @Agte506x;":CALC1:MARK1:FUNC:PPOL POS"
220 OUTPUT @Agte506x;":CALC1:MARK1:FUNC:EXEC"
230 OUTPUT @Agte506x;":CALC1:MARK1:X?"
240 ENTER @Agte506x;Freq
```

```
250 OUTPUT @Agte506x;":CALC1:MARK1:Y?"
260 ENTER @Agte506x;Resp
270 Img$="8A,MD.4DE,2X,MD.6DE"
280 PRINT " Frequency Response"
290 PRINT USING Img$;"Peak: ",Freq,Resp
300 !
310 PRINT "All Peaks Search using Command"
320 !
330 OUTPUT @Agte506x;":CALC1:FUNC:DOM OFF"
340 OUTPUT @Agte506x;":CALC1:FUNC:TYPE APE"
350 OUTPUT @Agte506x;":CALC1:FUNC:PEXC ";Excursion
360 OUTPUT @Agte506x;":CALC1:FUNC:PPOL POS"
370 OUTPUT @Agte506x;":CALC1:FUNC:EXEC"
380 OUTPUT @Agte506x;":CALC1:FUNC:POIN?"
390 ENTER @Agte506x;Poin
400 REDIM Result(1:Poin,1:2)
410 OUTPUT @Agte506x;":CALC1:FUNC:DATA?"
420 ENTER @Agte506x;Result(*)
430 Img$="4A,2D,2A,MD.4DE,2X,MD.6DE"
440 PRINT " Frequency Response"
450 FOR I=1 TO Poin
460 PRINT USING Img$;"Peak",I,": ",Result(I,2),Result(I,1)
470 NEXT I
480 GOTO No_err
490 Err: OFF INTR 7
500 OUTPUT @Agte506x;";:SYST:ERR?"
510 ENTER @Agte506x;Err_no,Err_msg$
520 PRINT "Error occurred!!"
530 PRINT " No:";Err_no,"Description: "&Err_msg$
540 No_err: OFF INTR 7
550 END
```

Limit Test

- Overview
- Sample Program in Excel VBA using VISA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to perform limit tests.

The sample program creates a limit table as shown in the following two tables, turns on the Limit Test feature, performs one cycle of measurement, and then displays the test results.

Limit table of trace 1

No.	Type	Begin Stimulus	End Stimulus	Begin Response	End Response
1	MAX	847.5 MHz	905.0 MHz	-55.0 dBm	-55.0 dBm
2	MIN	935.0 MHz	960.0 MHz	-3.5 dBm	-3.5 dBm
3	MAX	935.0 MHz	960.0 MHz	0 dBm	0 dBm
4	MAX	980.0 MHz	1047.5 MHz	-25.0 dBm	-25.0 dBm

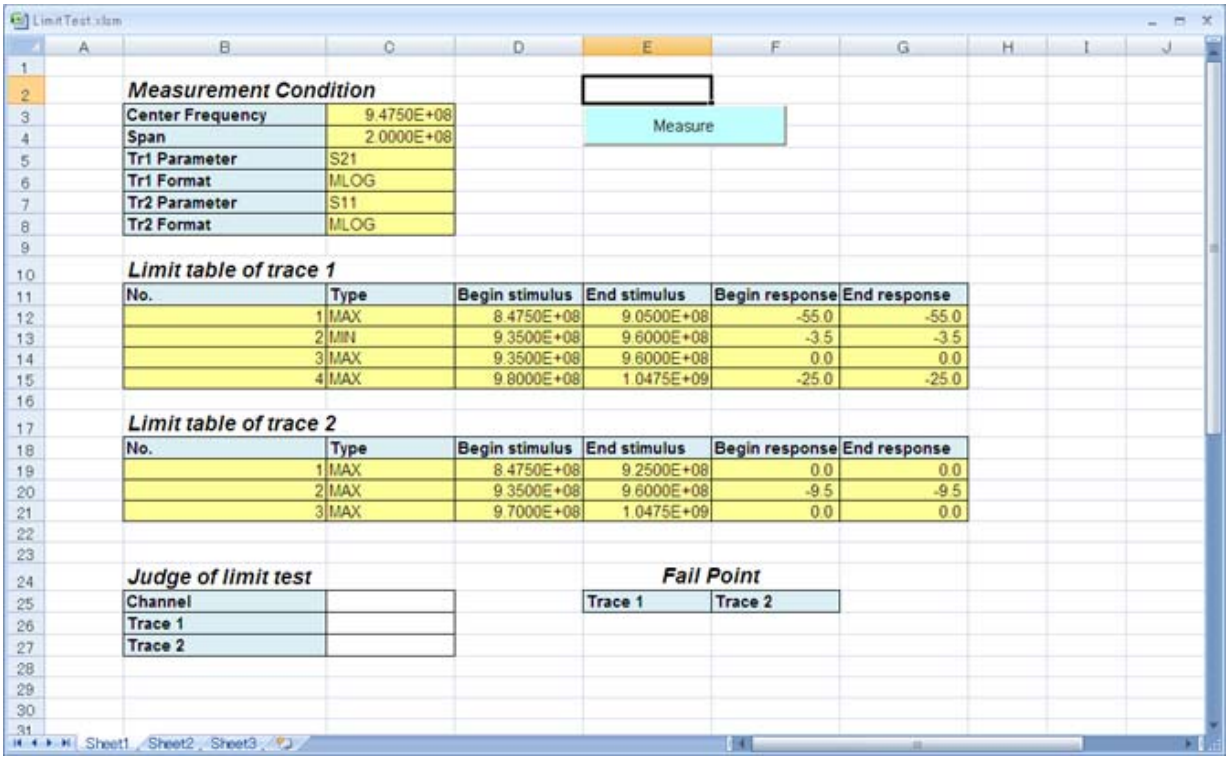
Limit table of trace 2

No.	Type	Begin Stimulus	End Stimulus	Begin Response	End Response
1	MAX	847.5 MHz	925.0 MHz	0 dBm	0 dBm
2	MAX	935.0 MHz	960.0 MHz	-9.5 dBm	-9.5 dBm
3	MAX	970.0 MHz	1047.5 MHz	0 dBm	0 dBm

See Limit Test for this programming.

Sample Program in Excel VBA using VISA

Example of excel sheet with limit test program



Measurement Condition

Center Frequency	9.4750E+08
Span	2.0000E+08
Tr1 Parameter	S21
Tr1 Format	MLOG
Tr2 Parameter	S11
Tr2 Format	MLOG

Measure

Limit table of trace 1

No.	Type	Begin stimulus	End stimulus	Begin response	End response
1	MAX	8.4750E+08	9.0500E+08	-55.0	-55.0
2	MIN	9.3500E+08	9.6000E+08	-3.5	-3.5
3	MAX	9.3500E+08	9.6000E+08	0.0	0.0
4	MAX	9.8000E+08	1.0475E+09	-25.0	-25.0

Limit table of trace 2

No.	Type	Begin stimulus	End stimulus	Begin response	End response
1	MAX	8.4750E+08	9.2500E+08	0.0	0.0
2	MAX	9.3500E+08	9.6000E+08	-9.5	-9.5
3	MAX	9.7000E+08	1.0475E+09	0.0	0.0

Judge of limit test

Channel	
Trace 1	
Trace 2	

Fail Point

Trace 1	Trace 2

```
Private Sub Measure_Click()
    Dim defrm As Long
    Dim Age506x As Long

    Dim Cent As Double, Span As Double
    Dim Param(1) As String, Fmt(1) As String
    Dim NumofSeg(1) As Integer

    Dim LimTbl1 As LimitTbl1
    Dim LimTbl2 As LimitTbl2

    Dim Dummy As String * 20
    Dim ret As Integer

    Dim Lim_Judge As Integer
    Dim Tr1_Judge As Integer
    Dim Tr2_Judge As Integer
    Dim Fail_Point As Integer
    Dim Fail_Data() As Double
```

```
Dim ptr As Long
Dim Fail_Point2 As Integer
Dim Fail_Data2() As Double
Dim ptr2 As Long

'***
'*** Open session.
'***

Call viOpenDefaultRM(defrm)
Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, Age506x)
Call viSetAttribute(Age506x, VI_ATTR_TMO_VALUE, 10000)

'***
'*** Set variable of measurement condition.
'***

Cent = CDbl(Cells(3, 3).Value)
Span = CDbl(Cells(4, 3).Value)
Param(0) = Trim(Cells(5, 3).Value)
Fmt(0) = Trim(Cells(6, 3).Value)
Param(1) = Trim(Cells(7, 3).Value)
Fmt(1) = Trim(Cells(8, 3).Value)

'***
'*** Set variable of limit tables.
'***

NumofSeg(0) = 4
NumofSeg(1) = 3

For i = 0 To NumofSeg(0) - 1
   With LimTbl1
      If Trim(Cells(12 + i, 3).Value) = "MAX" Then
         .Typ(i) = 1
      Else
         .Typ(i) = 2
```

```
        End If
        .BeginStim(i) = CDbl(Cells(12 + i, 4).Value)
        .EndStim(i) = CDbl(Cells(12 + i, 5).Value)
        .BeginResp(i) = CDbl(Cells(12 + i, 6).Value)
        .EndResp(i) = CDbl(Cells(12 + i, 7).Value)
    End With
Next i

For i = 0 To NumofSeg(1) - 1
    With LimTbl2
        If Trim(Cells(19 + i, 3).Value) = "MAX" Then
            .Typ(i) = 1
        Else
            .Typ(i) = 2
        End If
        .BeginStim(i) = CDbl(Cells(19 + i, 4).Value)
        .EndStim(i) = CDbl(Cells(19 + i, 5).Value)
        .BeginResp(i) = CDbl(Cells(19 + i, 6).Value)
        .EndResp(i) = CDbl(Cells(19 + i, 7).Value)
    End With
Next i

'***
'*** Send measurement condition to E5061B.
'***

Call viVPrintf(Age506x, ":SENS1:FREQ:CENT " + CStr(Cent) + vbLf, 0)
Call viVPrintf(Age506x, ":SENS1:FREQ:SPAN " + CStr(Span) + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:PAR1:COUN 2" + vbLf, 0)
Call viVPrintf(Age506x, ":DISP:WIND1:SPL D1_2" + vbLf, 0)

Call viVPrintf(Age506x, ":TRIG:SOUR BUS" + vbLf, 0)
Call viVPrintf(Age506x, ":INIT1:CONT ON" + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" & vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)

'***
```

```
'*** Send measurement parameter and format of trace 1 to E5061B.
'***

Call viVPrintf(Age506x, ":CALC1:PAR1:SEL" + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:PAR1:DEF " + Param(0) + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:FORM " + Fmt(0) + vbLf, 0)

'***
'*** Send limit table of trace 1 to E5061B.
'***

Call viVPrintf(Age506x, ":CALC1:LIM:DATA " + CStr(NumofSeg(0)), 0)
For i = 0 To NumofSeg(0) - 1
   With LimTbl1
      Call viVPrintf(Age506x, "," + CStr(.Typ(i)), 0)
      Call viVPrintf(Age506x, "," + CStr(.BeginStim(i)), 0)
      Call viVPrintf(Age506x, "," + CStr(.EndStim(i)), 0)
      Call viVPrintf(Age506x, "," + CStr(.BeginResp(i)), 0)
      If i = NumofSeg(0) - 1 Then
         Call viVPrintf(Age506x, "," + CStr(.EndResp(i)) + vbLf, 0)
      Else
         Call viVPrintf(Age506x, "," + CStr(.EndResp(i)), 0)
      End If
   End With
Next i
Call viVPrintf(Age506x, ":CALC1:LIM:DISP ON" + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:LIM ON" + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" & vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)
'***
'*** Send measurement parameter and format of trace 2 to E5061B.
'***

Call viVPrintf(Age506x, ":CALC1:PAR2:SEL" + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:PAR2:DEF " + Param(1) + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:FORM " + Fmt(1) + vbLf, 0)
```

```
'***
'*** Send limit table of trace 2 to E5061B.
'***
Call viVPrintf(Age506x, ":CALC1:LIM:DATA " + CStr(NumofSeg(1)), 0)
For i = 0 To NumofSeg(1) - 1
   With LimTbl2
      Call viVPrintf(Age506x, "," + CStr(.Typ(i)), 0)
      Call viVPrintf(Age506x, "," + CStr(.BeginStim(i)), 0)
      Call viVPrintf(Age506x, "," + CStr(.EndStim(i)), 0)
      Call viVPrintf(Age506x, "," + CStr(.BeginResp(i)), 0)
      If i = NumofSeg(1) - 1 Then
         Call viVPrintf(Age506x, "," + CStr(.EndResp(i)) + vbLf, 0)
      Else
         Call viVPrintf(Age506x, "," + CStr(.EndResp(i)), 0)
      End If
   End With
Next i
Call viVPrintf(Age506x, ":CALC1:LIM:DISP ON" + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:LIM ON" + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" & vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)

'***
'*** Setting status resister.
'***

Call viVPrintf(Age506x, ":STAT:QUES:LIM:CHAN1:ENAB 6" + vbLf, 0)
Call viVPrintf(Age506x, ":STAT:QUES:LIM:CHAN1:PTR 6" + vbLf, 0)
Call viVPrintf(Age506x, ":STAT:QUES:LIM:CHAN1:NTR 0" + vbLf, 0)
Call viVPrintf(Age506x, ":STAT:QUES:LIM:PTR 2" + vbLf, 0)
Call viVPrintf(Age506x, ":STAT:QUES:LIM:NTR 0" + vbLf, 0)
Call viVPrintf(Age506x, "*CLS" + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" & vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)

'***
'*** Trigger.
```

```
'***

Call viVPrintf(Age506x, ":TRIG:SING" + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" & vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)

'***
'*** Checking test results.
'***

Call viVPrintf(Age506x, ":STAT:QUES:LIM?" + vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)
ret = CInt(Dummy)
Lim_Judge = ret And 2
Call viVPrintf(Age506x, ":STAT:QUES:LIM:CHAN1?" + vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)
ret = CInt(Dummy)
Tr1_Judge = ret And 2
Tr2_Judge = ret And 4

'***
'*** Displaying test results.
'***

If Lim_Judge = 0 Then
   Cells(25, 3).Value = "PASS"
Else
   Cells(25, 3).Value = "FAIL"
   If Tr1_Judge = 0 Then
      Cells(26, 3).Value = "PASS"
   Else
      Cells(26, 3).Value = "FAIL"
      Call viVPrintf(Age506x, ":CALC1:PAR1:SEL" + vbLf, 0)
      Call viVPrintf(Age506x, ":CALC1:LIM:REP:POIN?" + vbLf, 0)
      Call viVScanf(Age506x, "%t", Dummy)
      Fail_Point = CInt(Dummy)
      ReDim Fail_Data(Fail_Point - 1)
```

```
            ptr = VarPtr(Fail_Data(0))
            Call viVPrintf(Age506x, ":CALC1:LIM:REP?" + vbLf, 0)
            Call viVScanf(Age506x, "%," + CStr(Fail_Point) + "lf", ptr)
            For i = 0 To Fail_Point - 1
               Cells(26 + i, 5).Value = Fail_Data(i)
            Next i
         End If
         If Tr2_Judge = 0 Then
            Cells(27, 3).Value = "PASS"
         Else
            Cells(27, 3).Value = "FAIL"
            Call viVPrintf(Age506x, ":CALC1:PAR2:SEL" + vbLf, 0)
            Call viVPrintf(Age506x, ":CALC1:LIM:REP:POIN?" + vbLf, 0)
            Call viVScanf(Age506x, "%t", Dummy)
            Fail_Point2 = CInt(Dummy)
            ReDim Fail_Data2(Fail_Point2 - 1)
            ptr2 = VarPtr(Fail_Data2(0))
            Call viVPrintf(Age506x, ":CALC1:LIM:REP?" + vbLf, 0)
            Call viVScanf(Age506x, "%," + CStr(Fail_Point2) + "lf", ptr2)
            For i = 0 To Fail_Point2 - 1
               Cells(26 + i, 6).Value = Fail_Data2(i)
            Next i
         End If
      End If
      Call viClose(Age506x)
      Call viClose(defrm)

End Sub

'************************
'*** The public user definition type variable can not be defined in the object module.
'*** It is necessary to write the following codes in standard module, etc.
'************************
Type LimitTbl1
   Typ(3) As Integer       '1:MAX 2:MIN
   BeginStim(3) As Double
   EndStim(3) As Double
```

```
    BeginResp(3) As Double
    EndResp(3) As Double
  End Type
  Type LimitTbl2
    Typ(2) As Integer       '1:MAX 2:MIN
    BeginStim(2) As Double
    EndStim(2) As Double
    BeginResp(2) As Double
    EndResp(2) As Double
  End Type
  '************************
```

Sample Program in HT Basic (lim_test.htb)

```
10 DIM Param1$[9],Param2$[9],Fmt1$[9],Fmt2$[9],Buff$[9]
20 REAL Cent,Span,Lim1(1:4,1:5),Lim2(1:3,1:5),Fail_data(1:1601)
30 INTEGER Num_of_seg1,Num_of_seg2,Segment,Column,Fail_point
40 !
50 ASSIGN @Agte506x TO 717
60 !
70 Cent=9.475E+8
80 Span=2.00E+8
90 Param1$="S21"
100 Param2$="S11"
110 Fmt1$="MLOG"
120 Fmt2$="MLOG"
130 !
140 ! == Trace 1 Limit Line ==
150 Num_of_seg1=4 ! Number of segments: 4
160 ! -- Segment 1 --
170 Lim1(1,1)=1 ! Type : Maximum
180 Lim1(1,2)=8.475E+8 ! Frequency Start: 847.5 MHz
190 Lim1(1,3)=9.050E+8 ! Stop : 905.0 MHz
200 Lim1(1,4)=-55 ! Response Start: -55 dBm
210 Lim1(1,5)=-55 ! Stop : -55 dBm
220 ! -- Segment 2 --
230 Lim1(2,1)=2 ! Type : Minimum
240 Lim1(2,2)=9.350E+8 ! Frequency Start: 935.0 MHz
250 Lim1(2,3)=9.600E+8 ! Stop : 960.0 MHz
```

```
260 Lim1(2,4)=-3.5 ! Response Start: -3.5 dBm
270 Lim1(2,5)=-3.5 ! Stop : -3.5 dBm
280 ! -- Segment 3 --
290 Lim1(3,1)=1 ! Type : Maximum
300 Lim1(3,2)=9.350E+8 ! Frequency Start: 935.0 MHz
310 Lim1(3,3)=9.600E+8 ! Stop : 960.0 MHz
320 Lim1(3,4)=0 ! Response Start: 0 dBm
330 Lim1(3,5)=0 ! Stop : 0 dBm
340 ! -- Segment 4 --
350 Lim1(4,1)=1 ! Type : Maximum
360 Lim1(4,2)=9.800E+8 ! Frequency Start: 980.0 MHz
370 Lim1(4,3)=1.0475E+9 ! Stop : 1047.5 MHz
380 Lim1(4,4)=-25 ! Response Start: -25 dBm
390 Lim1(4,5)=-25 ! Stop : -25 dBm
400 ! == Trace 2 Limit Line ==
410 Num_of_seg2=3 ! Number of segments: 3
420 ! -- Segment 1 --
430 Lim2(1,1)=1 ! Type : Maximum
440 Lim2(1,2)=8.475E+8 ! Frequency Start: 847.5 MHz
450 Lim2(1,3)=9.250E+8 ! Stop : 925.0 MHz
460 Lim2(1,4)=0 ! Response Start: 0 dBm
470 Lim2(1,5)=0 ! Stop : 0 dBm
480 ! -- Segment 2 --
490 Lim2(2,1)=1 ! Type : Maximum
500 Lim2(2,2)=9.350E+8 ! Frequency Start: 935.0 MHz
510 Lim2(2,3)=9.600E+8 ! Stop : 960.0 MHz
520 Lim2(2,4)=-9.5 ! Response Start: -9.5 dBm
530 Lim2(2,5)=-9.5 ! Stop : -9.5 dBm
540 ! -- Segment 3 --
550 Lim2(3,1)=1 ! Type : Maximum
560 Lim2(3,2)=9.700E+8 ! Frequency Start: 970.0 MHz
570 Lim2(3,3)=1.0475E+9 ! Stop : 1047.5 MHz
580 Lim2(3,4)=0 ! Response Start: 0 dBm
590 Lim2(3,5)=0 ! Stop : 0 dBm
600 !
610 OUTPUT @Agte506x;":SENS1:FREQ:CENT ";Cent
620 OUTPUT @Agte506x;":SENS1:FREQ:SPAN ";Span
```

```
630 OUTPUT @Agte506x;":CALC1:PAR1:COUN 2"
640 OUTPUT @Agte506x;":DISP:WIND1:SPL D1_2"
650 OUTPUT @Agte506x;":TRIG:SOUR BUS"
660 OUTPUT @Agte506x;":INIT1:CONT ON"
670 !
680 ! Trace 1
690 !
700 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
710 !
720 OUTPUT @Agte506x;":CALC1:PAR1:DEF "&Param1$
730 OUTPUT @Agte506x;":CALC1:FORM "&Fmt1$
740 !
750 OUTPUT @Agte506x;":CALC1:LIM:DATA ";Num_of_seg1;
760 FOR Segment=1 TO Num_of_seg1
770 FOR Column=1 TO 5
780 OUTPUT @Agte506x;",";Lim1(Segment,Column);
790 NEXT Column
800 NEXT Segment
810 OUTPUT @Agte506x;""
820 OUTPUT @Agte506x;":CALC1:LIM:DISP ON"
830 OUTPUT @Agte506x;":CALC1:LIM ON"
840 !
850 ! Trace 2
860 !
870 OUTPUT @Agte506x;":CALC1:PAR2:SEL"
880 !
890 OUTPUT @Agte506x;":CALC1:PAR2:DEF "&Param2$
900 OUTPUT @Agte506x;":CALC1:FORM "&Fmt2$
910 !
920 OUTPUT @Agte506x;":CALC1:LIM:DATA ";Num_of_seg2;
930 FOR Segment=1 TO Num_of_seg2
940 FOR Column=1 TO 5
950 OUTPUT @Agte506x;",";Lim2(Segment,Column);
960 NEXT Column
970 NEXT Segment
980 OUTPUT @Agte506x;""
990 OUTPUT @Agte506x;":CALC1:LIM:DISP ON"
```

```
1000 OUTPUT @Agte506x;":CALC1:LIM ON"
1010 !
1020 ! Setting status registers
1030 !
1040 OUTPUT @Agte506x;":STAT:QUES:LIM:CHAN1:ENAB 6"
1050 OUTPUT @Agte506x;":STAT:QUES:LIM:CHAN1:PTR 6"
1060 OUTPUT @Agte506x;":STAT:QUES:LIM:CHAN1:NTR 0"
1070 OUTPUT @Agte506x;":STAT:QUES:LIM:PTR 2"
1080 OUTPUT @Agte506x;":STAT:QUES:LIM:NTR 0"
1090 OUTPUT @Agte506x;"*CLS"
1100 !
1110 OUTPUT @Agte506x;":TRIG:SING"
1120 OUTPUT @Agte506x;"*OPC?"
1130 ENTER @Agte506x;Buff$
1140 !
1150 ! Checking test results
1160 !
1170 OUTPUT @Agte506x;":STAT:QUES:LIM?"
1180 ENTER @Agte506x;Reg_val
1190 Ch1_judge=BIT(Reg_val,1)
1200 OUTPUT @Agte506x;":STAT:QUES:LIM:CHAN1?"
1210 ENTER @Agte506x;Reg_val
1220 Tr1_judge=BIT(Reg_val,1)
1230 Tr2_judge=BIT(Reg_val,2)
1240 !
1250 ! Displaying test results
1260 !
1270 IF Ch1_judge=0 THEN
1280 PRINT "## PASS! ##"
1290 ELSE
1300 PRINT "## FAIL! ##"
1310 IF Tr1_judge=0 THEN
1320 PRINT " Trace1(S21): PASS"
1330 ELSE
1340 PRINT " Trace1(S21): FAIL"
1350 !
1360 ! Reading and displaying frequency at failed points
```

```
1370 !
1380 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
1390 OUTPUT @Agte506x;":CALC1:LIM:REP:POIN?"
1400 ENTER @Agte506x;Fail_point
1410 REDIM Fail_data(1:Fail_point)
1420 OUTPUT @Agte506x;":CALC1:LIM:REP?"
1430 ENTER @Agte506x;Fail_data(*)
1440 PRINT " Frequency:"
1450 FOR I=1 TO Fail_point
1460 PRINT USING "3X,MD.4DE";Fail_data(I)
1470 NEXT I
1480 END IF
1490 IF Tr2_judge=0 THEN
1500 PRINT " Trace2(S11): PASS"
1510 ELSE
1520 PRINT " Trace2(S11): FAIL"
1530 !
1540 ! Reading and displaying frequency at failed points
1550 !
1560 OUTPUT @Agte506x;":CALC1:PAR2:SEL"
1570 OUTPUT @Agte506x;":CALC1:LIM:REP:POIN?"
1580 ENTER @Agte506x;Fail_point
1590 REDIM Fail_data(1:Fail_point)
1600 OUTPUT @Agte506x;":CALC1:LIM:REP?"
1610 ENTER @Agte506x;Fail_data(*)
1620 PRINT " Frequency:"
1630 FOR I=1 TO Fail_point
1640 PRINT USING "3X,MD.4DE";Fail_data(I)
1650 NEXT I
1660 END IF
1670 END IF
1680 END
```

Description

Line 50

Assigns a GPIB address to the I/O pass.

Lines 70 to 120

These lines store the sweep center value, sweep span value, trace 1 measurement parameter, trace 2 measurement parameter, trace 1 data format, and trace 2 data format into the variables Cent, Span, Param1$, Param2$, Fmt1$, and Fmt2$, respectively.

Line 150

Stores the number of segments in trace 1 limit table into the Num_of_seg1 variable.

Lines 160 to 390

These lines store the settings in trace 1 limit table into the Lim1(*) variable.

Line 410

Stores the number of segments in trace 2 limit table into the Num_of_seg2 variable.

Lines 420 to 590

These lines store the settings in trace 2 limit table into the Lim2(*) variable.

Lines 610 to 620

These lines configure the sweep range for channel 1's sweep range using the center and span values contained in the Cent and Span values.

Lines 630 to 660

These lines configure channel 1 so that it contains 2 traces, displays graphs in two windows tiled horizontally (i.e., with the screen split into upper and lower halves), uses a bus trigger source, and works in continuous activation mode.

Line 700

Sets channel 1's active trace to trace 1.

Lines 720 to 730

These lines store trace 1's measurement parameter and data format into the variables Param1$ and Fmt1$, respectively.

Lines 750 to 810

These lines set up the limit table for trace 1.

Line 750: Sends the command that sets up a limit table along with the Num_of_seg1 variable that contains the number of segments.

Lines 770 to 790: Sends five data items (type, start point stimulus value, end point stimulus value, start point response value, and end point response value) for each segment.

Lines 820 to 830

These lines turn ON the display of limit lines and the Limit Test feature for trace 1.

Line 870

Sets channel 1's active trace to trace 2.

Lines 890 to 900

These lines set trace 2's measurement parameter and data format to Param2$ and Fmt2$, respectively.

Lines 920 to 980

These lines set up the limit table for trace 2.

Lines 990 to 1000

These lines turn ON the display of limit lines and the Limit Test feature for trace 2.

Lines 1040 to 1060

These lines set, under the questionable limit channel 1 status register, the enable register and positive transition filter to 6 (0000000000000110 in binary notation) while setting the negative transition filter to 0 so that the questionable limit status condition register's bit 1 is set to 1 when the test result that combines the results for trace 1 and trace 2 is "fail."

The sample program provides an example of explicitly configuring the register bits so that they reflect the test results that only cover trace 1 and trace 2. However, because the results for traces 3 to 9 will never "fail" as long as the Limit Test feature is disabled for those traces, the register bits would reflect the test result that is limited to traces 1 and 2, even if the default setting is not changed.

Lines 1070 to 1080

These lines set transition filters so that the questionable limit status event register's bit 1 is set to 1 when the questionable limit status condition register's bit 1 changes from 0 to 1.

Line 1090

Clears the questionable limit status event register and questionable limit channel 1 status event register.

Lines 1110 to 1130

These lines trigger the instrument and wait until the sweep cycle is completed.

Lines 1170 to 1190

These lines retrieve the value of the questionable limit status event register and store the setting of bit 1 of the value into Ch1_judge.

Lines 1200 to 1230

These lines retrieve the value of the questionable limit channel 1 status event register and store the settings of bit 1 and bit 2 of the value into Tr1_judge and Tr2_judge, respectively.

Line 1280

Displays a message indicating that the DUT has passed the limit test if the test result for channel 1 is "Pass" (i.e., if Ch1_judge returns 0).

Lines 1300 to 1660

These lines are executed if the test result for channel 1 is "Fail" (i.e., if Ch1_judge returns 1).

Line 1300: Notifies the user that the limit test result is "Fail".

Line 1320: Displays a message indicating that trace 1 has passed the limit test if the test result for trace 1 is "Pass" (i.e., if Tr1_judge returns 0).

Lines 1340 to 1470: These lines are executed if the test result for trace 1 is "Fail" (i.e., if Tr1_judge returns 1). The lines notify the user that the test result for trace 1 is "Fail" and then retrieve and display the frequencies at the failed measurement points on trace 1.

Line 1340: Notifies the user that the limit test result for trace 1 is "Pass."

Line 1380: Sets channel 1's active trace to trace 2.

Lines 1390 to 1410: These lines retrieve the number of failed measurement points on trace 1 and, based on that number, resize the array that contains retrieved frequencies.

Lines 1420 to 1470: These lines retrieve and display the frequencies at the failed measurement points on trace 1.

Line 1500: Displays a message indicating that trace 2 has passed the limit test if the test result for trace 2 is "Pass" (i.e., if Tr2_judge returns 0).

Lines 1520 to 1650: If the test result for trace 2 is "Fail" (i.e., if Tr2_judge returns 1), these lines notify the user that trace 2 has failed to pass the limit test and then retrieve and display the frequencies at the failed measurement points on trace 2.

Bandwidth Search

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

The sample program demonstrates how to perform Bandwidth Search.

The sample program moves the marker to the maximum value position and then retrieves and displays the results of Bandwidth Search.

Sample Program in Excel VBA

```
Sub Bandwid_Click()
    Dim vi As Long
    Dim defrm As Long
    Dim Threshhold As Long
    Dim Result As String * 1000
    Dim Bdata As Variant

    Dim Dummy As String * 20

    Const TimeOutTime = 10000

    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)

    Threshhold = -3
    Call viVPrintf(vi, ":SENS1:FREQ:CENT 947.5E6" & vbLf, 0)
    Call viVPrintf(vi, ":SENS1:FREQ:SPAN 200E6" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:PAR1:DEF S21" & vbLf, 0)
    Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:PAR1:SEL" & vbLf, 0)
    Call viVPrintf(vi, ":INIT1:CONT ON" & vbLf, 0)
    Call viVPrintf(vi, ":TRIG:SOUR BUS" & vbLf, 0)
    Call viVPrintf(vi, "*OPC?" & vbLf, 0)
    Call viVScanf(vi, "%t", Dummy)

    Call viVPrintf(vi, ":TRIG:SING" & vbLf, 0)
```

```
    Call viVPrintf(vi, "*OPC?" & vbLf, 0)
    Call viVScanf(vi, "%t", Dummy)

    Call viVPrintf(vi, ":CALC1:MARK1 ON" & vbLf, 0)

    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:TYPE MAX" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:FUNC:EXEC" & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK1:BWID:THR " + CStr(Threshhold) & vbLf, 0)
    Call viVPrintf(vi, ":CALC1:MARK:BWID ON" & vbLf, 0)
    Call viVPrintf(vi, "*OPC?" & vbLf, 0)
    Call viVScanf(vi, "%t", Dummy)

    Call viVPrintf(vi, ":CALC1:MARK1:BWID:DATA?" & vbLf, 0)
    Call viVScanf(vi, "%t", Result)
    Bdata = Split(Result, ",")
    Call ErrorCheck(vi)
    Cells(5, 2).Value = Val(Bdata(0))
    Cells(6, 2).Value = Val(Bdata(1))
    Cells(7, 2).Value = Val(Bdata(2))
    Cells(8, 2).Value = Val(Trim(Bdata(3)))

    Call viClose(vi)
    Call viClose(defrm)
End Sub
Sub ErrorCheck(vi)
    Dim err As String * 50, ErrNo As Variant, Response
    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)
    ErrNo = Split(err, ",")
    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)
    End If
End Sub
```

Sample Program in HT Basic (bandwid.htb)

```
10 DIM Buff$[9],Err_msg$[100]
20 REAL Threshold,Bwid,Cent,Q,Loss
30 INTEGER Err_no
```

```
40 !
50 ASSIGN @Agte506x TO 717
60 Threshold=-3
70 !
80 OUTPUT @Agte506x;"*ESE 60"
90 OUTPUT @Agte506x;"*SRE 32"
100 OUTPUT @Agte506x;"*CLS"
110 OUTPUT @Agte506x;"*OPC?"
120 ENTER @Agte506x;Buff$
130 ON INTR 7 GOTO Err
140 ENABLE INTR 7;2
150 !
160 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
170 OUTPUT @Agte506x;":CALC1:MARK1:FUNC:TYPE MAX"
180 OUTPUT @Agte506x;":CALC1:MARK1:FUNC:EXEC"
190 OUTPUT @Agte506x;":CALC1:MARK1:BWID:THR ";Threshold
200 OUTPUT @Agte506x;":CALC1:MARK1:BWID:DATA?"
210 WAIT .5
220 ENTER @Agte506x;Bwid,Cent,Q,Loss
230 !
240 PRINT "## Bandwidth Search ##"
250 PRINT "Bandwidth : ",Bwid
260 PRINT "Center Frequency: ",Cent
270 PRINT "Q : ",Q
280 PRINT "Loss : ",Loss
290 !
300 GOTO No_err
310 Err: OFF INTR 7
320 OUTPUT @Agte506x;";:SYST:ERR?"
330 ENTER @Agte506x;Err_no,Err_msg$
340 PRINT "Error occurred!!"
350 PRINT " No:";Err_no,"Description: "&Err_msg$
360 No_err: OFF INTR 7
370 END
```

Saving Files

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

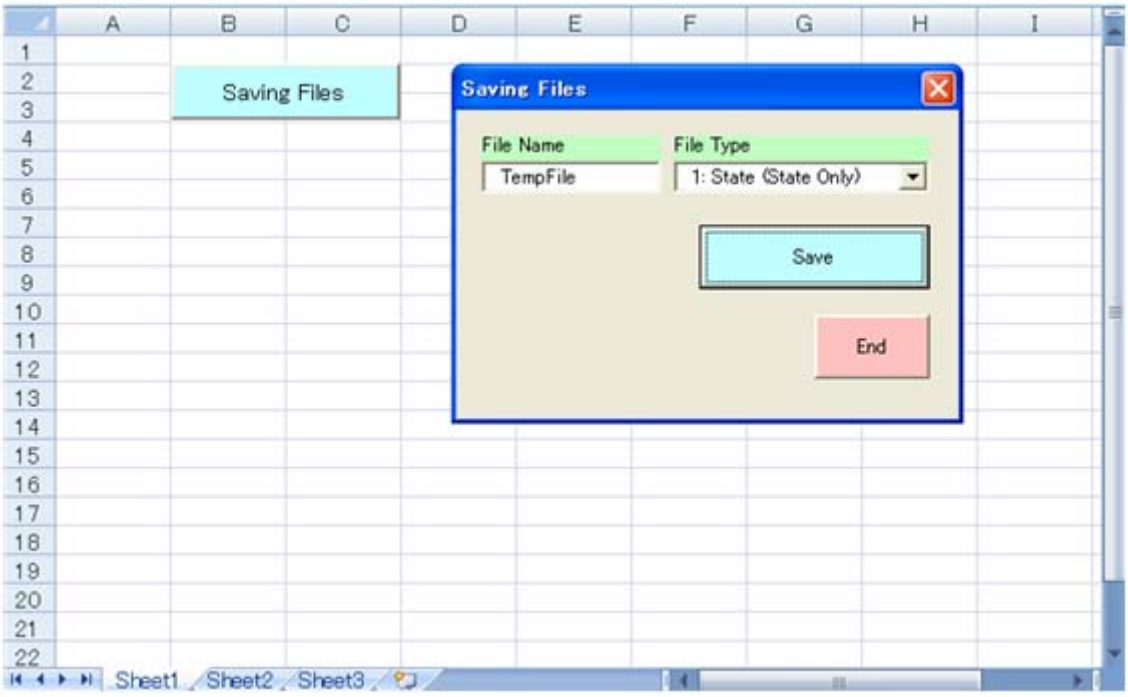
Overview

The sample program demonstrates how to save a file. This program saves selected content on a file with a specified name.

See Saving and Recalling File for this programming.

Sample Program in Excel VBA

Example of excel sheet and window form with saving files program




```
Private Sub File_Save_Click()
    ' Declare two string variables for file name and file type
    Dim File_Name As String
    Dim File_Type As String
    Dim defrm As Long
    Dim vi As Long
    Const TimeOutTime = 10000
    ' Check whether file name textbox is empty or not
    If TextBox1.Text <> "" Then
```

```
        File_Name = Trim(TextBox1.Text)
        File_Type = Trim(frmFileSave.ComboBox1.Value)
    ' Open connection to the E5061B
        Call viOpenDefaultRM(defrm)
        Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)
        Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)
        Select Case File_Type
            Case "1: State (State Only)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP STAT" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" + vbLf, 0)
            Case "2: State (State & Cal)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP CST" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" + vbLf, 0)
            Case "3: State (State & Trace)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP DST" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" + vbLf, 0)
            Case "4: State (All)"
                Call viVPrintf(vi, ":MMEM:STOR:STYP CDST" + vbLf, 0)
                Call viVPrintf(vi, ":MMEM:STOR """ & File_Name & ".sta""" + vbLf, 0)
            Case "5: Trace Data (CSV)"
                Call viVPrintf(vi, ":MMEM:STOR:FDAT """ & File_Name & ".csv""" + vbLf, 0)
            Case "6: Screen Image (BMP)"
                Call viVPrintf(vi, ":MMEM:STOR:IMAG """ & File_Name & ".bmp""" + vbLf, 0)
            Case Else
                MsgBox "Error in code"
        End Select
        Call viClose(defrm)
    Else
        MsgBox "Please enter a filename"
    End If
End Sub

Private Sub UserForm_Initialize()
  ComboBox1.AddItem "1: State (State Only)"
  ComboBox1.AddItem "2: State (State & Cal)"
  ComboBox1.AddItem "3: State (State & Trace)"
  ComboBox1.AddItem "4: State (All)"
```

```
  ComboBox1.AddItem "5: Trace Data (CSV)"
  ComboBox1.AddItem "6: Screen Image (BMP)"

  ComboBox1.ListIndex = 0

  TextBox1.Text = "TempFile"
End Sub

Private Sub EndBtn_Click()
  End
End Sub
```

Sample Program in HT Basic (file_sav.htb)

```
10 DIM File$[300],Inp_char$[30]
20 INTEGER Content
30 CLEAR SCREEN
40 ASSIGN @Agte506x TO 717
50 !
60 ON ERROR GOTO Content_select
70 Content_select: !
80 PRINT "## Save Content Selection ##"
90 PRINT "Select Content"
100 PRINT " 1: State (State only)"
110 PRINT " 2: State (State & Cal)"
120 PRINT " 3: State (State & Trace)"
130 PRINT " 4: State (State & Cal & Trace)"
140 PRINT " 5: Trace Data (CSV)"
150 PRINT " 6: Screen"
160 PRINT ""
170 PRINT "Input 1 to 6"
180 INPUT "Number?",Inp_char$
190 Content=IVAL(Inp_char$,10)
200 IF Content<1 OR Content>6 THEN Content_select
210 OFF ERROR
220 !
230 CALL Inp_file_name(File$)
240 !
```

```
250 SELECT Content
260 CASE 1
270 OUTPUT @Agte506x;":MMEM:STOR:STYP STAT"
280 OUTPUT @Agte506x;":MMEM:STOR """&File$&".sta"""
290 CASE 2
300 OUTPUT @Agte506x;":MMEM:STOR:STYP CST"
310 OUTPUT @Agte506x;":MMEM:STOR """&File$&".sta"""
320 CASE 3
330 OUTPUT @Agte506x;":MMEM:STOR:STYP DST"
340 OUTPUT @Agte506x;":MMEM:STOR """&File$&".sta"""
350 CASE 4
360 OUTPUT @Agte506x;":MMEM:STOR:STYP CDST"
370 OUTPUT @Agte506x;":MMEM:STOR """&File$&".sta"""
380 CASE 5
390 OUTPUT @Agte506x;":MMEM:STOR:FDAT """&File$&".csv"""
400 CASE 6
410 OUTPUT @Agte506x;":MMEM:STOR:IMAG """&File$&".bmp"""
420 END SELECT
430 !
440 END
450 !=========================================
460 ! File Name Input Function
470 !=========================================
480 SUB Inp_file_name(Inp_name$)
490 DIM Inp_char$[9]
500 ON ERROR GOTO Inp_start
510 Inp_start: !
520 PRINT "## File Name Input ##"
530 PRINT "Input Save File Name (without Extension)"
540 INPUT "Name?",Inp_name$
550 PRINT "Input Name: "&Inp_name$
560 INPUT "OK? [Y/N]",Inp_char$
570 IF UPC$(Inp_char$)<>"Y" THEN Inp_start
580 OFF ERROR
590 SUBEND
```

Transferring Files

- Overview
- Sample Program in Excel VBA using VISA
- Sample Program in Excel VBA using VISA-COM
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

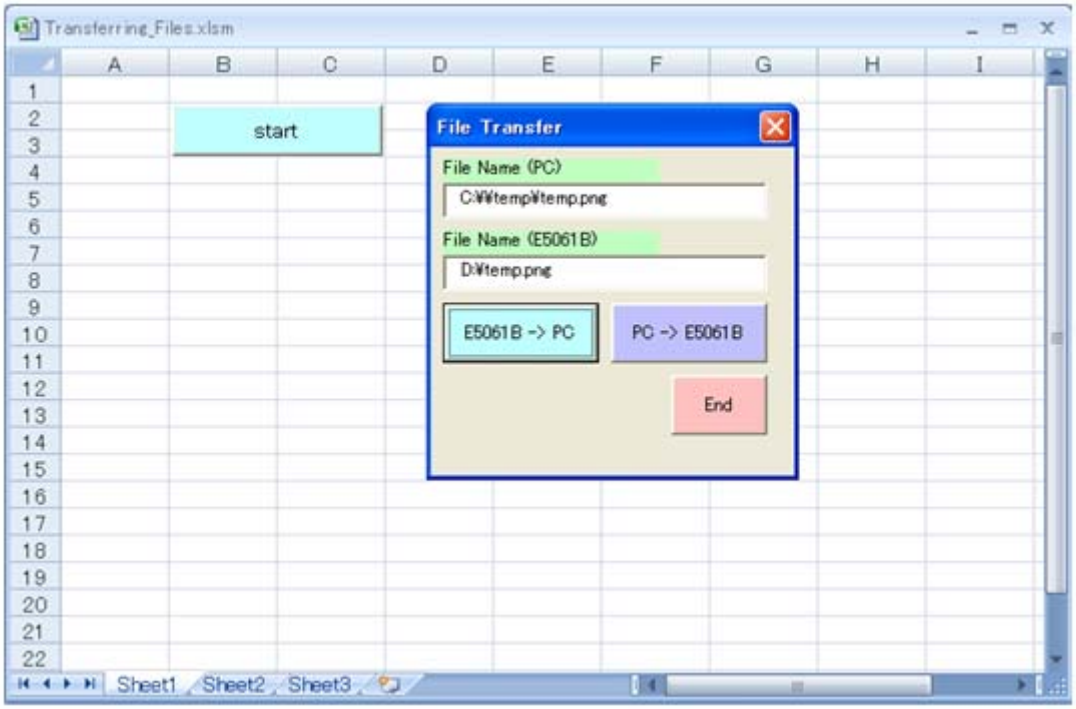
This sample program demonstrates hw to transfer files between the external controller and the E5061B.

This program reads out data from a specified file on the external controller (or the E5061B), then writes them to a specified file on the E5061B (or the external controller).

See Managing Files for this programming.

Sample Program in Excel VBA using VISA

Example of excel sheet and window form with transferring files program




```
Private Sub fromE5061B_toPC_Click()
    '*** This sequence is a sample code in which the file is transferred
```

```
'*** from the E5061B to the external controller.
Dim defrm As Long
Dim Age506x As Long
Dim Res As Variant

'*** The maximum size of transferring file is 5000000 byte.
Dim byteData(5000000) As Byte
Dim PtrData(1) As Long
Dim RetCnt As Long
Dim workData() As Byte
Dim hFile As Long
Dim isOpen As Boolean

Dim E5061B_File As String
Dim PC_File As String

PtrData(0) = VarPtr(RetCnt)
PtrData(1) = VarPtr(byteData(0))
RetCnt = UBound(byteData) - LBound(byteData) + 1

E5061B_File = """" & Trim(TextBox2.Text) & """"
PC_File = TextBox1.Text

Call viOpenDefaultRM(defrm)
Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, Age506x)
Call viSetAttribute(Age506x, VI_ATTR_TMO_VALUE, 10000)
Call viVPrintf(Age506x, ":MMEM:TRAN? " + E5061B_File + vbLf, 0)

Call viVScanf(Age506x, "%#b", PtrData(0))

ReDim workData(RetCnt - 1)

For i = 0 To RetCnt - 2
  workData(i) = byteData(i)
Next i

hFile = FreeFile()
```

```
    Open PC_File For Binary Access Write Shared As hFile
    isOpen = True

    Put #hFile, , workData
    If isOpen Then Close #hFile

    Call viClose(Age506x)
End Sub

Private Sub fromPC_toE5061B_Click()
    '*** This sequence is a sample code in which the file is transferred
    '*** from the external controller to the E5061B.
    Dim hFile As Long
    Dim isOpen As Boolean
    Dim byteData() As Byte
    Dim strBuf As String
    Dim fileSize As Long
    Dim PtrData As Long

    Dim defrm As Long
    Dim Age506x As Long

    Dim E5061B_File As String
    Dim PC_File As String

    E5061B_File = """" & Trim(TextBox2.Text) & """"
    PC_File = TextBox1.Text

    fileSize = FileLen(PC_File)
    ReDim byteData(fileSize - 1)
    PtrData = VarPtr(byteData(0))
    hFile = FreeFile()
    Open PC_File For Binary Access Read Shared As hFile
    isOpen = True

    Get #hFile, , byteData
    If isOpen Then Close #hFile
```

```
    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, Age506x)
    Call viSetAttribute(Age506x, VI_ATTR_TMO_VALUE, 10000)
    strBuf = ":MMEM:TRAN " & E5061B_File & "," & "%#" & CLng(fileSize - 1) & "b" & vbLf
    Call viVPrintf(Age506x, strBuf, PtrData)

    Call viClose(Age506x)
End Sub

Private Sub UserForm_Initialize()
  TextBox1.Text = "C:\\temp\temp.png"
  TextBox2.Text = "D:\temp.png"
End Sub

Private Sub EndBtn_Click()
    End
End Sub
```

Sample Program in Excel VBA using VISA-COM

```
Private Sub fromE5061B_toPC_Click()
    '*** This sequence is a sample code in which the file is transferred
    '*** from the E5061B to the external controller.
    Dim hFile As Long
    Dim isOpen As Boolean
    Dim ioMgr As VisaComLib.ResourceManager
    Set ioMgr = New VisaComLib.ResourceManager

    Dim Age506x As VisaComLib.FormattedIO488
    Set Age506x = New VisaComLib.FormattedIO488

    Set Age506x.IO = ioMgr.Open("GPIB0::17::INSTR")
    Age506x.IO.Timeout = 10000

    Dim byteData() As Byte
    Dim E5061B_File As String
    Dim PC_File As String
```

```
    E5061B_File = """" & Trim(TextBox2.Text) & """"
    PC_File = Trim(TextBox1.Text)

    Age506x.WriteString ":MMEM:TRAN? " & E5061B_File
    byteData = Age506x.ReadIEEEBlock(BinaryType_UI1)

    hFile = FreeFile()
    Open PC_File For Binary Access Write Shared As hFile
    isOpen = True

    Put #hFile, , byteData
    If isOpen Then Close #hFile
    Age506x.IO.Close
End Sub

Private Sub fromPC_toE5061B_Click()
    '*** This sequence is a sample code in which the file is transferred
    '*** from the external controller to the E5061B.
    Dim hFile As Long
    Dim isOpen As Boolean
    Dim ioMgr As VisaComLib.ResourceManager
    Set ioMgr = New VisaComLib.ResourceManager

    Dim Age506x As VisaComLib.FormattedIO488
    Set Age506x = New VisaComLib.FormattedIO488

    Set Age506x.IO = ioMgr.Open("GPIB0::17::INSTR")
    Age506x.IO.Timeout = 10000

    Dim byteData() As Byte
    Dim strBuf As String
    Dim fileSize As Long
    Dim E5061B_File As String
    Dim PC_File As String

    E5061B_File = """" & Trim(TextBox2.Text) & """"
    PC_File = Trim(TextBox1.Text)
```

```
    fileSize = FileLen(PC_File)
    ReDim byteData(fileSize - 1)
    hFile = FreeFile()
    Open PC_File For Binary Access Read Shared As hFile
    isOpen = True

    Get #hFile, , byteData
    If isOpen Then Close #hFile
    strBuf = ":MMEM:TRAN " & E5061B_File & ","
    Age506x.WriteIEEEBlock strBuf, byteData, True

    Age506x.IO.Close
End Sub

Private Sub UserForm_Initialize()
  TextBox1.Text = "C:\\temp\temp.png"
  TextBox2.Text = "D:\temp.png"
End Sub

Private Sub EndBtn_Click()
  End
End Sub
```

Sample Program in HT Basic (file_xfr.htb)

```
10 DIM Src_file$[50],Dst_file$[50],Src_size_char$[50],Inp_char$[30]
20 INTEGER Direction
30 ASSIGN @Agte506x TO 717
40 !
50 CLEAR SCREEN
60 ON ERROR GOTO Direct_select
70 Direct_select: !
80 PRINT "#### File Transfer ####"
90 PRINT " 1: E506x -> Controller"
100 PRINT " 2: Controller -> E506x"
110 PRINT ""
120 PRINT "Input 1 or 2"
```

```
130 INPUT "Number?",Inp_char$
140 Direction=IVAL(Inp_char$,10)
150 IF Direction<1 OR Direction>2 THEN Direct_select
160 OFF ERROR
170 !
180 PRINT ""
190 PRINT " Input source file name. ";
200 INPUT "Name?",Src_file$
210 PRINT ": "&Src_file$
220 !
230 IF Direction=2 THEN
240 PRINT " Input source file size. ";
250 INPUT "Size[Byte]?",Src_size_char$
260 PRINT ": "&Src_size_char$&"[Byte]"
270 END IF
280 !
290 PRINT " Input destination file name. ";
300 INPUT "Name?",Dst_file$
310 PRINT ": "&Dst_file$
320 PRINT ""
330 !
340 IF Direction=1 THEN
350 Copy_to_contr(@Agte506x,Src_file$,Dst_file$)
360 ELSE
370 Copy_to_e506x(@Agte506x,Src_file$,Src_size_char$,Dst_file$)
380 END IF
390 !
400 END
410 !=========================================
420 ! File Transfer Function (E506x -> Controller)
430 !=========================================
440 SUB Copy_to_contr(@Agte506x,Src_file$,Dst_file$)
450 DIM Img$[32],Src_size_char$[10],Buff$[9],Err_msg$[100]
460 INTEGER Max_bsize,Block_size,Err_no
470 REAL Src_size
480 !
490 ON ERROR GOTO Skip_purge
```

```
500 PURGE Dst_file$
510 Skip_purge: OFF ERROR
520 CREATE Dst_file$,1
530 ASSIGN @Dst_file TO Dst_file$
540 Max_bsize=24576 ! 24KByte
550 !
560 OUTPUT @Agte506x;"*ESE 60"
570 OUTPUT @Agte506x;"*SRE 32"
580 OUTPUT @Agte506x;"*CLS"
590 OUTPUT @Agte506x;"*OPC?"
600 ENTER @Agte506x;Buff$
610 !
620 ON INTR 7 GOTO Err
630 ENABLE INTR 7;2
640 PRINT "Now Copying: "&Src_file$&"(@E506x) -> "&Dst_file$&"(@Contro ller)"
650 OUTPUT @Agte506x;":MMEM:TRAN? """&Src_file$&""""
660 WAIT .1
670 ENTER @Agte506x USING "#,A";Buff$
680 ENTER @Agte506x USING "#,A";Digit$
690 Img$="#,"&Digit$&"A"
700 ENTER @Agte506x USING Img$;Src_size_char$
710 !
720 Src_size=VAL(Src_size_char$)
730 WHILE Src_size>0
740 IF Src_size>Max_bsize THEN
750 Block_size=Max_bsize
760 ELSE
770 Block_size=Src_size
780 END IF
790 !
800 ALLOCATE Dat$[Block_size]
810 Img$="#,"&VAL$(Block_size)&"A"
820 ENTER @Agte506x USING Img$;Dat$
830 OUTPUT @Dst_file USING Img$;Dat$
840 DEALLOCATE Dat$
850 !
860 Src_size=Src_size-Block_size
```

```
870 END WHILE
880 !
890 PRINT "Done"
900 ENTER @Agte506x USING "#,A";Buff$
910 ASSIGN @Dst_file TO *
920 !
930 GOTO Skip_error
940 Err: OFF INTR 7
950 OUTPUT @Agte506x;";:SYST:ERR?"
960 ENTER @Agte506x;Err_no,Err_msg$
970 PRINT "Error occurred!!"
980 PRINT " No:";Err_no,"Description: "&Err_msg$
990 Skip_error: OFF INTR 7
1000 SUBEND
1010 !==========================================
1020 ! File Transfer Function (Controller -> E506x)
1030 !==========================================
1040 SUB Copy_to_e506x(@Agte506x,Src_file$,Src_size_char$,Dst_file$)
1050 DIM Img$[32],Header$[10],Buff$[9],Err_msg$[100]
1060 INTEGER Max_bsize,Block_size,Err_no
1070 REAL Src_size
1080 !
1090 ON ERROR GOTO File_error
1100 ASSIGN @Src_file TO Src_file$
1110 OFF ERROR
1120 Max_bsize=24576 ! 24KByte
1130 !
1140 OUTPUT @Agte506x;"*CLS"
1150 OUTPUT @Agte506x;"*OPC?"
1160 ENTER @Agte506x;Buff$
1170 !
1180 PRINT "Now Copying: "&Src_file$&"(@Controller) -> "&Dst_file$&"(@ E506x)"
1190 Header$="#"&VAL$(LEN(Src_size_char$))&Src_size_char$
1200 OUTPUT @Agte506x;":MMEM:TRAN """&Dst_file$&""","&Header$;
1210 !
1220 Src_size=VAL(Src_size_char$)
1230 WHILE Src_size>0
```

```
1240 IF Src_size>Max_bsize THEN
1250 Block_size=Max_bsize
1260 ELSE
1270 Block_size=Src_size
1280 END IF
1290 !
1300 ALLOCATE Dat$[Block_size]
1310 Img$="#,"&VAL$(Block_size)&"A"
1320 ENTER @Src_file USING Img$;Dat$
1330 OUTPUT @Agte506x USING Img$;Dat$
1340 DEALLOCATE Dat$
1350 !
1360 Src_size=Src_size-Block_size
1370 END WHILE
1380 !
1390 OUTPUT @Agte506x;"",END
1400 ASSIGN @Src_file TO *
1410 !
1420 OUTPUT @Agte506x;";:SYST:ERR?"
1430 ENTER @Agte506x;Err_no,Err_msg$
1440 IF Err_no=0 THEN
1450 PRINT "Done"
1460 ELSE
1470 PRINT "Error occurred!!"
1480 PRINT " No:";Err_no,"Description: "&Err_msg$
1490 END IF
1500 GOTO Skip_error
1510 File_error:OFF ERROR
1520 PRINT "File name NOT found!"
1530 Skip_error:!
1540 SUBEND
```

Description

Line 40

Assigns a GPIB address to the I/O pass.

Lines 60 to 130

These line allow the user to return to the entry start line and re-enter the data if an error (such as an invalid entry) occurs while entering the number that indicates the transfer direction. Then, these line display the list of transfer directions and prompt the user to input a selected number.

Lines 80 to 130

These lines display the list of transfer directions, and prompt the user to choose one of the items by typing in the appropriate number.

Lines 140 to 150

Converts the entered value into an integer and stores it into the Direction variable. Returns to the entry start line if an invalid value is contained in Direction.

Lines 180 to 210

These lines obtain the name of the source file for copying from the user input, store it into the Src_file$ variable, and display the value of Src_file$.

Lines 180 to 210

These lines obtain the name of the source file for copying from the user input, store it into the Src_file$ variable, and display the value of Src_file$.

Lines 230 to 270

If Direction is equal to 2 (from the external controller to the E5061B), these lines obtain the size of the source file for copying, store it into the Src_size_char$, and display the value of Src_size_char$.

Lines 290 to 320

These lines obtain the name of the destination file for copying from the user input, store it into the Dst_file$ variable, and display the value of Dst_file$.

Line 350

If Direction is equal to 1 (from the E5061B to the external controller), these lines use the subprogram Copy_to_contr to transfer (copy) a file with the name Src_file$ on the E5061B to a file with the name Dst_file$ on the external controller.

Line 370

If Direction is equal to 2, these lines use the subprogram Copy_to_e506x to transfer (copy) a file with the name Src_file$ on the external controller to a file with the name Dst_file$ on the E5061B.

Copy_to_contr, a subprogram for transferring files from the E5061B to the external controller that appears in lines 440 to 1000, is described below.

Lines 490 to 520

If any file with the name File$ already exists, these lines delete the file and create a new file with the name File$.

Line 530

Assigns a destination file for copying to the I/O pass.

Line 540

This line stores a maximum number of transferred data (in bytes) per one transfer, that is 24 KByte to meet the size limitation of string arrays in the HTBasic, into Max_bsize variable.

Lines 560 to 600

These lines configure the system to generate an SRQ when it cannot find a source file for copying due to an error.

Lines 620 to 630

These lines set the branch target for an SRQ interrupt to enable SRQ interrupts.

Lines 640 to 650

These lines display a message showing that the transfer has started, and execute commands for reading data from a file on the E5061B.

Lines 670 to 680

These lines read the header symbol (#) in a block data, read number of digits (characters) indicating the size of data in bytes, then store it into Digit$ variable.

Line 690

This line creates a format for reading characters in Digit$.

Line 700

This line reads the data size in byte and stores it into Src_size_char$ variable.

Line 720

This line converts Src_size_char$ to a real number and stores it into Src_size variable.

Lines 730 to 870

These lines repeat the procedures below until Src_size reaches 0.

Lines 740 to 780: If Src_size is greater than Max_bsize, these lines assign the value of the Max_bsize to Block_size variable (transferred data in bytes). If Src_size is equal or less than Max_bsize, assign the value of Src_size to Block_size.

Line 800 This line defines Dat$ string variable with the size as large as Block_size and reserves memory area.

Line 810 This line creates a format for reading characters as many as Block_size characters.

Line 820 This line reads data from the file on the E5061B, then stores them into Dat$.

Line 830 This line writes the contents of Dat$ to the file on the external controller.

Lines 840 to 860 These lines free the memory area for Dat$ and subtract Block_size from Src_size.

Lines 890 to 900

These lines display a message showing the completion of transfer, then read a message terminator at the end of the data.

Lines 940 to 980

These lines define an error handler that retrieves and displays the number and message of an error that has occurred.

Copy_to_e506x, a subprogram for transferring files from the external controller to the E5061B that appears in lines 1040 to 1540, is described below.

Lines 1090 to 1110

Assigns a destination file for copying to the I/O pass.

Line 1120

This line stores a maximum number of transferred data (in bytes) per one transfer, that is 24 KByte, into Max_bsize variable.

Lines 1140 to 1160

Clears the error queue.

Line 1180

Displays a measurement start message.

Lines 1190 to 1200

These lines create the header part indicating that data will be sent as many as Src_size_char$ bytes, then send the header part of the command and its parameters for writing the data to the file on the E5061B.

Line 1220

This line converts Src_size_char$ to a real number and stores it into Src_size variable.

Lines 1230 to 1370

These lines repeat the procedures below until Src_size reaches 0.

Lines 1240 to 1280: If Src_size is greater than Max_bsize, these lines assign the value of the Max_bsiize to Block_size variable (transferred data in bytes). If Src_size is equal or less than Max_bsize, assign the value of Src_size to Block_size.

Line 1300 This line defines Dat$ string variable with the size as large as Block_size and reserves memory area.

Line 1310 This line creates a format for reading characters as many as Block_size characters.

Line 1320 This line reads data from the file on the external controller, then stores them into Dat$.

Line 1330 This line writes the contents of Dat$ to the file on the E5061B.

Lines 1340 to 1360 These lines free the memory area for Dat$ and subtract Block_size from Src_size.

Line 1390

This line sends a message terminator at the end of data.

Lines 1420 to 1430

These lines retrieve the error number and error message from the error queue, and then store them into the variables Err_no and Err_msg$, respectively.

Lines 1440 to 1490

If Err_no is equal to 0 (no error occurred), these lines display the message indicating completion of transfer, and if Err_no is not equal to 0 (an error occurred), display Err_no along with Err_msg$.

Lines 1510 to 1520

These lines handle the case with no source file for copying is found.

Time Domain

- Overview
- Sample Program in Excel VBA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

This sample program demonstrates how to use the transformation function of the time domain function.

This program executes calibration (ECal), performs measurement once, converts the results to data in the time domain, and displays this data.

See Analysis in Time Domain (time domain function) for this programming.

Sample Program in Excel VBA

```
Private Sub Time_Domain_Click()
    Dim defrm As Long                   'Session to Default Resource Manager
    Dim vi As Long                      'Session to instrument
    Dim Para As String
    Dim Tran_Type As String
    Dim Stim_Type As String
    Dim stop_freq As Double
    Dim Win_Beta As Double
    Dim Star_Time As Double
    Dim Stop_Time As Double
    Dim Result As String

    Const TimeOutTime = 40000           'timeout time.

    stop_freq = 3000000000#             'Stop Frequendy  : 3 GHz
    Nop = 201                           'Nop             : 201
    Para = "S11"                        'Meas. Parameter : S11
    Tran_Type = "LPAS"                  'Transform Type  : Lowpass
    Stim_Type = "IMP"                   'Stimulus Type   : Impulse
    Win_Beta = 13                       'Window Beta     : 13 (Maximum Type)
    Star_Time = 0                       'Start time      : 0 sec
    Stop_Time = 0.00000001              'Stop time       : 10 nsec

    Call viOpenDefaultRM(defrm)                                 'Initializes the VISA system.
```

```
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)                    'Opens the session to the
specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)             'The state of an attribute
for the specified session.

    Call viVPrintf(vi, ":SYST:PRES" & vbLf, 0)                          'Presets the setting state of the ENA.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)                                'Clears the all status register.

    Call viVPrintf(vi, ":SENS1:FREQ:STOP " & stop_freq & vbLf, 0)       'Sets the sweep stop
frequency.
    Call viVPrintf(vi, ":SENS1:SWE:POIN " & Nop & vbLf, 0)              'Sets the number of points.

    Call viVPrintf(vi, ":CALC1:TRAN:TIME:LPFR" & vbLf, 0)               'Sets a measurement point.

    Call viVPrintf(vi, ":CALC1:PAR1:DEF " & Para & vbLf, 0)             'Sets the measurement
parameter.
    Call viVPrintf(vi, ":TRIG:SOUR BUS" & vbLf, 0)                      'Sets the trigger source to BUS.
    Call ErrorCheck(vi)                                                 'Checking the error.

    MsgBox "Connect Port1 to Ecal Module. Then click OK button.", vbOKOnly      'Display the message
box.

    Call viVPrintf(vi, ":SENS1:CORR:COLL:ECAL:SOLT1 1" & vbLf, 0)       'Execute the 1-port
calibration.
    Call viVPrintf(vi, "*OPC?" & vbLf, 0)                               'Sets the *OPC? command.
    Call viVScanf(vi, "%t", Result)                                     'Reads the *OPC? result.
    Call ErrorCheck(vi)                                                 'Checking the error.
    MsgBox "Set DUT. Then click OK button.", vbOKOnly                   'Display the message box.

    Call viVPrintf(vi, ":TRIG:SING" & vbLf, 0)                          'Execute the trigger.
    Call viVPrintf(vi, "*OPC?" & vbLf, 0)                               'Sets the *OPC? command.
    Call viVScanf(vi, "%t", Result)                                     'Reads the *OPC? result.
    Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)            'Execute auto scale.

    MsgBox "Click OK button. < Time Domain Transform >", vbOKOnly       'Display the message
box.

    Call viVPrintf(vi, ":CALC1:TRAN:TIME " & Tran_Type & vbLf, 0)       'Sets the transformation
type.
```

```
    Call viVPrintf(vi, ":CALC1:TRAN:TIME:STIM " & Stim_Type & vbLf, 0)          'Sets the stimulus type.
    Call viVPrintf(vi, ":CALC1:TRAN:TIME:KBES " & Win_Beta & vbLf, 0)          'Sets the beta value of
the window.
    Call viVPrintf(vi, ":CALC1:TRAN:TIME:STAR " & Star_Time & vbLf, 0)         'Sets the start value of
the display range.
    Call viVPrintf(vi, ":CALC1:TRAN:TIME:STOP " & Stop_Time & vbLf, 0)         'Sets the end value of
the display range.
    Call viVPrintf(vi, ":CALC1:TRAN:TIME:STAT ON" & vbLf, 0)                   'Turns on the time domain
function.
    Call ErrorCheck(vi)                                                        'Checking the error.

    Call viVPrintf(vi, ":CALC1:PAR1:SEL" & vbLf, 0)                            'Sets the active trace.
    Call viVPrintf(vi, ":CALC1:FORM REAL" & vbLf, 0)                           'Sets the real data format.
    Call viVPrintf(vi, ":DISP:WIND1:TRAC1:Y:AUTO" & vbLf, 0)                   'Execute auto scale.
    Call ErrorCheck(vi)                                                        'Checking the error.

    Call viClose(vi)                                                           'Closes the resource manager session.
    Call viClose(defrm)                                                        'Breaks the communication and terminates
the VISA system.

    End                                                                        'End
End Sub

Private Sub EndBtn_Click()
    End
End Sub

Sub ErrorCheck(vi)
    Dim err As String * 50, ErrNo As Variant, Response As VbMsgBoxResult
    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)
    ErrNo = Split(err, ",")
    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)
    End If
End Sub
```

Sample Program in HT Basic

```
10 DIM Para$[9],Tran_type$[9],Stim_type$[9],Buff$[9],Inp_ch ar$[9]
20 REAL Stop_freq,Win_beta,Star_time,Stop_time
```

```
30 INTEGER Nop
40 !
50 ASSIGN @Agte506x TO 717
60 !
70 Stop_freq=3.E+9 ! Stop Frequency : 3 GHz
80 Nop=201 ! Nop : 201
90 Para$="S11" ! Meas. Parameter : S11
100 !
110 Tran_type$="LPAS" ! Transform Type : Lowpass
120 Stim_type$="IMP" ! Stimulus Type : Impulse
130 Win_beta=13 ! Window Beta : 13 (Maximum Type)
140 Star_time=0 ! Start time : 0 s
150 Stop_time=1.E-8 ! Stop time : 10 ns
160 !
170 OUTPUT @Agte506x;":SYST:PRES"
180 OUTPUT @Agte506x;":SENS1:FREQ:STOP ";Stop_freq
190 OUTPUT @Agte506x;":SENS1:SWE:POIN ";Nop
200 !
210 OUTPUT @Agte506x;":CALC1:TRAN:TIME:LPFR"
220 !
230 OUTPUT @Agte506x;":CALC1:PAR1:DEF "&Para$
240 OUTPUT @Agte506x;":TRIG:SOUR BUS"
250 !
260 ! 1 Port Full Calibration (ECal)
270 !
280 PRINT "Connect Port 1 to ECal Module. Then push [Enter] key."
290 INPUT "",Buff$
300 OUTPUT @Agte506x;":SENS1:CORR:COLL:ECAL:SOLT1 1"
310 OUTPUT @Agte506x;":SYST:ERR?"
320 ENTER @Agte506x;Buff$
330 !
340 ! Measurement
350 !
360 PRINT "Set DUT. Then Push [Enter] key."
370 INPUT "",Inp_char$
380 !
390 OUTPUT @Agte506x;":TRIG:SING"
```

```
400 OUTPUT @Agte506x;"*OPC?"
410 ENTER @Agte506x;Buff$
420 !
430 OUTPUT @Agte506x;":DISP:WIND1:TRAC1:Y:AUTO"
440 PRINT "Push [Enter] key. -> [Time Domain Transform]"
450 INPUT "",Inp_char$
460 !
470 ! Time Domain Transform
480 !
490 OUTPUT @Agte506x;":CALC1:TRAN:TIME "&Tran_type$
500 OUTPUT @Agte506x;":CALC1:TRAN:TIME:STIM "&Stim_type$
510 OUTPUT @Agte506x;":CALC1:TRAN:TIME:KBES ";Win_beta
520 OUTPUT @Agte506x;":CALC1:TRAN:TIME:STAR ";Star_time
530 OUTPUT @Agte506x;":CALC1:TRAN:TIME:STOP ";Stop_time
540 OUTPUT @Agte506x;":CALC1:TRAN:TIME:STAT ON"
550 !
560 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
570 OUTPUT @Agte506x;":CALC1:FORM REAL"
580 OUTPUT @Agte506x;":DISP:WIND1:TRAC1:Y:AUTO"
590 END
```

Control Using SICL-LAN Server

- Overview
- Sample Program in Excel VBA

Other topics about Sample Application Program

Overview

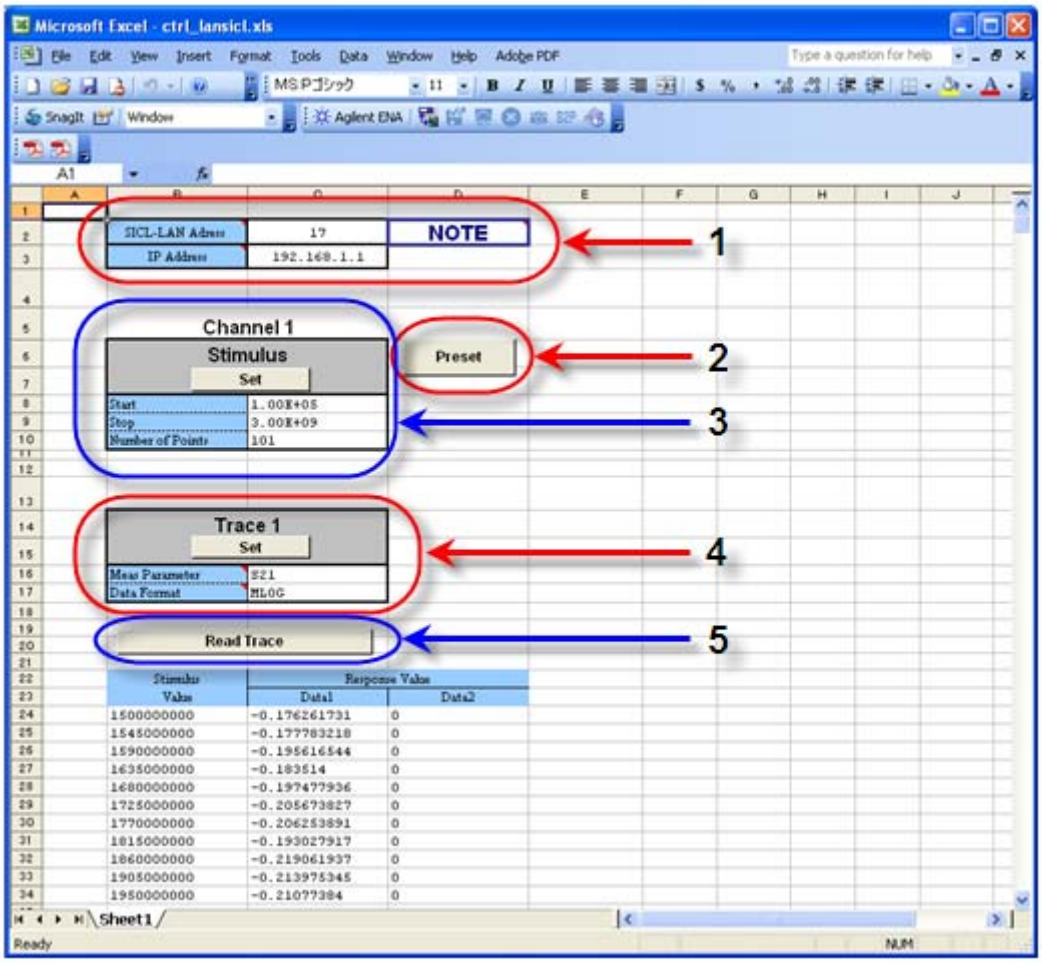
This section explains how to control the E5061B by using SICL in the Windows environment.

NOTE To control the E5061B using the SICL-LAN server, you need to make the preparations described in Control over SICL-LAN server.

Sample Program in Excel VBA

Opening **ctrl_lansicl.xls** in Microsoft Excel displays a screen as shown in the figure below:

ctrl_lansicl.xls



For how to use each element in ctrl_lansicl.xls, refer to the following description.

1. In part 1, in the cell to the right of the SICL-LAN Address, enter the address of the E5061B for control with the SICL-LAN server. This address is **XX**, which has been set with the command **System** > **Misc Setup** > **Network Setup** > **SICL-LAN Address** [**XX**]. Enter the IP address of the E5061B in the cell to the right of the IP Address. This VBA macro will not work properly without the correct values in these two cells.
2. Click Preset in part 2 to execute the presetting operation.
3. In part 3, the sweep range (start and stop points) and the number of measurement points for channel 1 are set. Click **Set** to execute the setting as shown in the setting table.
4. Part 4 sets the measurement parameters and data format for trace 1 in channel 1. Click **Set** to execute the setting as shown in the setting table.
5. Click **Read Trace** in part 5 retrieves the formatted data array of trace 1 in channel 1. The data is displayed in tabular.

Description of Operation in VBA macro

This section describes the operation of the VBA macro, focusing on the part related to control with SICL.

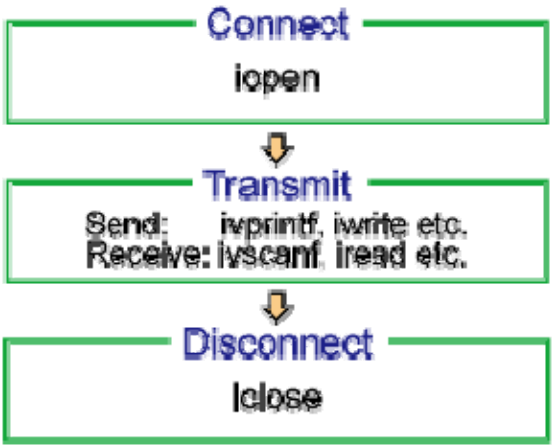
NOTE In order to use SICL in your VBA macro, you must declare functions and define variables with a SICL definition file (for VB).

NOTE In the VBA macro, ctrl_sicllan.xls, the standard module whose object name is "SICL," is the definition file**.**

The basic control flow with SICL is shown in Flow of control using SICL.

NOTE In this sample program, the **ivprintf** function, the **ivscanf** function, and the **iread** function are used in its communication part; you can use other SICL functions as well. For details, refer to sicl.hlp (the online help of SICL).

Flow of control using SICL



e5071c441

NOTE For more information on how to use each function of SICL, refer to the SICL manual.

The procedures of each step in Flow of control using SICL are described below.

Connection

The procedure corresponding to connection is OpenSession (OpenSession). OpenSession establishes a connection to the E5061B with the **iopen** function of SICL, using the SICL-LAN Address and IP Address entered in part 1 in ctrl_lansicl.xls. The **iopen** function takes the address information of the E5061B you specify as its parameters.

Syntax

addr = iopen(*dev*)

Variable	**addr**
Description	Session information (output)
Data type	Integer type
	dev
Description	Address information of the instrument you specify (input)
Data type	Character string type

Grammar	*sicl-name [ip-address]:interface, sicl-lan-address*

For example, if the parameter (*dev*) is "lan[192.168.0.1]:hpib9,17," connection is made to the address of **17** of the interface of **hpib9** with the E5061B whose IP address is **192.168.0.1** by using the external controller whose SICL interface name is **lan**.

OpenSession

```
Function OpenSession() As Integer
Dim ServAddr As String
Dim IpAddr As String
On Error GoTo ErrHandler
'"Get Sicl-Lan Address
Sheets("Sheet1").Select
Range("C2").Select
ServAddr = ActiveCell.FormulaR1C1
'"Get Ip Address
Sheets("Sheet1").Select
Range("C3").Select
IpAddr = ActiveCell.FormulaR1C1

OpenSession = iopen("lan[" & IpAddr & "]:hpib9," & ServAddr)
Call itimeout(OpenSession, 10000)
Exit Function
ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
End
End Function
```

Sending

The procedure corresponding to sending in communication is OutputSiclLan. OutputSiclLan uses the **ivprintf** function of SICL to send messages (SCPI commands). The **ivprintf** function takes the session information output from the **iopen** function and a program message as its parameters.

Syntax

Status = ivprintf(*addr,mes*)

Variable	**Status**
Description	Return value of the function (output)
Data type	Integer type
	addr
Description	Session information (input)
Data type	Integer type
	mes
Description	Program message (input)
Data type	Character string type

OutputSiclLan

```
Sub OutputSiclLan(addr As Integer, message As String)

Dim Status As Integer
Dim actualcnt As Long
Dim length As Long

On Error GoTo ErrHandler
length = Len(message)
Status = ivprintf(addr, message & Chr$(10))
Exit Sub

ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
```

```
End
End Sub
```

Receiving

The procedure corresponding to receiving ASCII format messages in communication is EnterSiclLan. EnterSiclLan uses the **ivscanf** function of SICL to receive a message in ASCII format and store it into the output variable. The **ivscanf** function takes the session information output from the **iopen** function, the format for output, and the data to be output as its parameters.

Syntax

Status = ivscanf(*addr,fmt,ap*)

Variable	**fmt**
Description	Format for output (input)
Data type	Character string type
	ap
Description	Data to be output (output)
Data type	Character string type

For information on the variable (*Status*) and the variable (*addr*), refer to Variable.

In Visual Basic, variables must be declared as a fixed-length string when receiving string data using the **ivscanf** function.

EnterSiclLan

```
Sub EnterSiclLan(addr As Integer, Query As String)

Dim Status As Integer
Dim actualcnt As Long
```

```
Dim res As String * 256

On Error GoTo ErrHandler
Status = ivscanf(addr, "%t", res)
Query = Trim(res)
Exit Sub

ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
End

End Sub
```

The procedure corresponding to receiving array data in communication is EnterSiclLanArrayReal64, which uses the **iread** function of SICL to receive array data in the IEEE 64-bit floating point binary transfer format and store it into the output variable. The **iread** function takes the session information output from the **iopen** function, the data to be output, the number of data bytes, the condition to finish reading data, and the number of data bytes actually read out as its parameters.

Syntax

Status = iread(*addr,buf,bufsize,reason,actual*)

Variable	**buf**
Description	Data to be output (output)
Data type	Character string type
	bufsize
Description	The number of data bytes (input)
Data type	Long integer type
	reason
Description	The condition to finish reading out data (input)

Data type	Integer type
	actual
Description	The number of data bytes actually read out (output)
Data type	Long integer type

For information on the variable (*Status*) and the variable (*addr*), refer to Variable.

Each functional of EnterSiclLanArrayReal64 is described below.

(1) Retrieves the data header.

(2) Stores the number of data bytes into the size variable in the header part.

(3) Retrieves the formatted data array for trace 1 in channel 1 and stores it into the databuf variable.

(4) Retrieves the message terminator at the end of the data.

EnterSiclLanArrayReal64

Function EnterSiclLanArrayReal64(addr As Integer, databuf() As Double) As Long

```
Dim Status As Integer
Dim actualcnt As Long
Dim buf As String * 8
Dim size As Long
On Error GoTo ErrHandler
'"Read header info of "#6NNNNNN"
Status = iread(addr, buf, 8, I_TERM_MAXCNT, actualcnt) '.....(1)
size = Val(Mid$(buf, 3, 6)) '.....(2)
'"Read data
Status = iread(addr, databuf, size, I_TERM_MAXCNT, actualcnt) '.....(3)
'"Read ending LF
Status = iread(addr, buf, 1, I_TERM_MAXCNT, actualcnt) '.....(4)
EnterSiclLanArrayReal64 = size / 8
```

```
Exit Function

ErrHandler:
MsgBox "*** Error : " & Error$
Call siclcleanup
End
End Function
```

Disconnection

The **iclose** function of SICL is used to disconnect communication. The **iclose** function takes the session information output from the **iopen** function as its parameter.

Syntax

Status = iclose(*addr*)

For information on the variable (*Status*) and the variable (*addr*), refer to Variable

Sample control

The E5061B can be controlled by executing the above procedures in order, following the control flow in Flow of control using SICL. This is demonstrated by the Preset procedure (a procedure that is executed when the Preset button is clicked) as described in Preset.

Preset

```
Sub Preset()

''' Open Session
E506x = OpenSession

'''Presetting the analyzer
Call OutputSiclLan(E506x, ":SYST:PRES")
'''Close Session
Call iclose(E506x)

End Sub
```

Control Using Telnet Server

- Overview
- Sample Program in Excel VBA

Other topics about Sample Programs

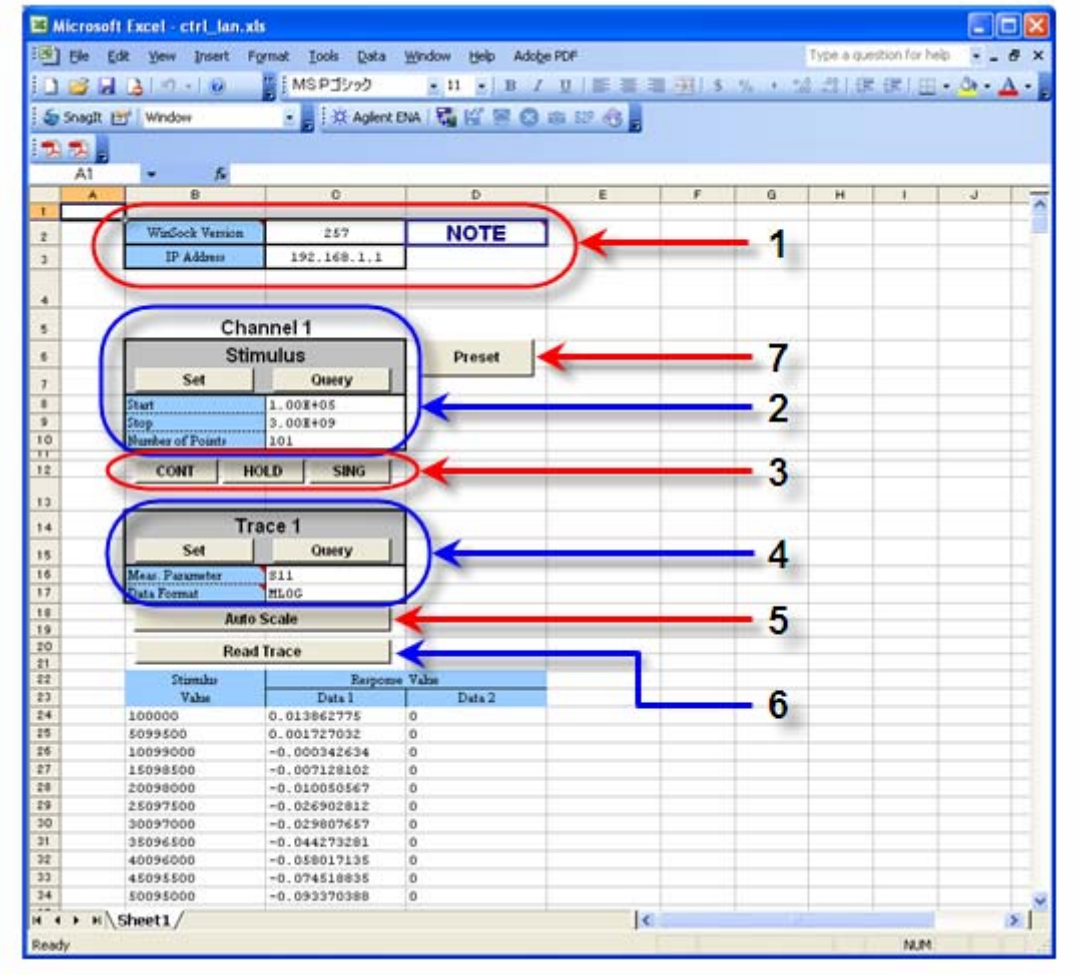
Overview

This section explains how to control the E5061B by using WinSock API in the Windows environment.

Sample Program in Excel VBA

Opening **ctrl_lan.xls** in Microsoft Excel displays the screen shown in the figure below.

ctrl_lan.xls



For how to use each element in ctrl_lan.xls, refer to the following description.

1. Enter the version number of WinSock API in the cell to the right side of "Winsock Version." The version number is obtained by multiplying

256 by the major version and then adding the minor version. For example, when the version of your Winsock API is 1.1, the version number is obtained as follows: 256×1+1=257. Enter the IP address of the E5061B in the cell to the right side of "IP Address." This VBA macro will not work properly without the correct values in these two cells.

2. In part 2, the sweep range (start and stop points) and the number of measurement points are set. Click **Set** to execute the setting operation as specified with the setting table, while clicking the button labeled "Query" retrieves the current settings of the E5061B.
3. Part 3 is dedicated to setting the trigger mode.
4. Part 4 sets the measurement parameters and data format for trace 1 in channel 1. Click **Set** to execute the setting operation as specified with the setting table, while clicking the button labeled "Query" retrieves the current settings of the E5061B.
5. In part 5, click **Auto Scale** to execute auto scaling for trace 1 in channel 1.
6. Click **Read Trace** in part 6 to retrieve the formatted data of trace 1 in channel 1. The data is displayed in tabular.
7. Click **Preset** to execute the presetting operation.

Description of operation in VBA macro

This section describes the operation of the VBA macro, focusing on the part related to control with WinSock API.

In order to use WinSock API, you must declare functions and define variables with a definition file of WinSock API, as shown in Definition file of WinSock API.

Definition file of WinSock API

```
'This is the Winsock API definition file for Visual Basic


'Setup the variable type 'hostent' for the WSAStartup command
Type Hostent
h_name As Long
h_aliases As Long
h_addrtype As String * 2
h_length As String * 2
h_addr_list As Long
End Type
```

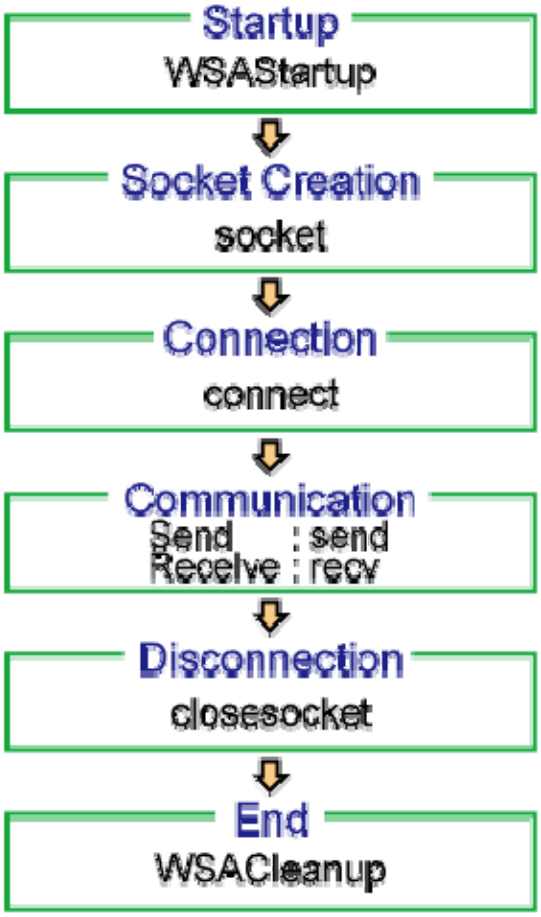
```
Public Const SZHOSTENT = 16

'Set the Internet address type to a long integer (32-bit)
Type in_addr
s_addr As Long
End Type

'A note to those familiar with the C header file for Winsock
'Visual Basic does not permit a user-defined variable type
'to be used as a return structure. In the case of the
'variable definition below, sin_addr must
'be declared as a long integer rather than the user-defined
'variable type of in_addr.
Type sockaddr_in
sin_family As Integer
sin_port As Integer
sin_addr As Long
sin_zero As String * 8
End Type

Public Const WSADESCRIPTION_LEN = 256
Public Const WSASYS_STATUS_LEN = 128
Public Const WSA_DescriptionSize = WSADESCRIPTION_LEN + 1
Public Const WSA_SysStatusSize = WSASYS_STATUS_LEN + 1

'Setup the structure for the information returned from
'the WSAStartup() function.
Type WSAData
wVersion As Integer
wHighVersion As Integer
szDescription As String * WSA_DescriptionSize
szSystemStatus As String * WSA_SysStatusSize
iMaxSockets As Integer
iMaxUdpDg As Integer
lpVendorInfo As String * 200
End Type
```

```
'Define socket return codes
Public Const INVALID_SOCKET = &HFFFF
Public Const SOCKET_ERROR = -1

'Define socket types
Public Const SOCK_STREAM = 1 'Stream socket
Public Const SOCK_DGRAM = 2 'Datagram socket
Public Const SOCK_RAW = 3 'Raw data socket
Public Const SOCK_RDM = 4 'Reliable Delivery socket
Public Const SOCK_SEQPACKET = 5 'Sequenced Packet socket

'Define address families
Public Const AF_UNSPEC = 0 'unspecified
Public Const AF_UNIX = 1 'local to host (pipes, portals)
Public Const AF_INET = 2 'internetwork: UDP, TCP, etc.
Public Const AF_IMPLINK = 3 'arpanet imp addresses
Public Const AF_PUP = 4 'pup protocols: e.g. BSP
Public Const AF_CHAOS = 5 'mit CHAOS protocols
Public Const AF_NS = 6 'XEROX NS protocols
Public Const AF_ISO = 7 'ISO protocols
Public Const AF_OSI = AF_ISO 'OSI is ISO
Public Const AF_ECMA = 8 'european computer manufacturers
Public Const AF_DATAKIT = 9 'datakit protocols
Public Const AF_CCITT = 10 'CCITT protocols, X.25 etc
Public Const AF_SNA = 11 'IBM SNA
Public Const AF_DECnet = 12 'DECnet
Public Const AF_DLI = 13 'Direct data link interface
Public Const AF_LAT = 14 'LAT
Public Const AF_HYLINK = 15 'NSC Hyperchannel
Public Const AF_APPLETALK = 16 'AppleTalk
Public Const AF_NETBIOS = 17 'NetBios-style addresses
Public Const AF_MAX = 18 'Maximum # of address families

'Setup sockaddr data type to store Internet addresses
Type sockaddr
sa_family As Integer
sa_data As String * 14
```

```
End Type
Public Const SADDRLEN = 16

'Declare Socket functions

Public Declare Function closesocket Lib "wsock32.dll" (ByVal s As Long) As Long

Public Declare Function connect Lib "wsock32.dll" (ByVal s As Long, addr As sockaddr_in, ByVal
namelen As Long) As Long

Public Declare Function htons Lib "wsock32.dll" (ByVal hostshort As Long) As Integer

Public Declare Function inet_addr Lib "wsock32.dll" (ByVal cp As String) As Long

Public Declare Function recv Lib "wsock32.dll" (ByVal s As Long, ByVal buf As Any, ByVal buflen As
Long, ByVal flags As Long) As Long

Public Declare Function recvB Lib "wsock32.dll" Alias "recv" (ByVal s As Long, buf As Any, ByVal buflen
As Long, ByVal flags As Long) As Long

Public Declare Function send Lib "wsock32.dll" (ByVal s As Long, buf As Any, ByVal buflen As Long,
ByVal flags As Long) As Long

Public Declare Function socket Lib "wsock32.dll" (ByVal af As Long, ByVal socktype As Long, ByVal
protocol As Long) As Long

Public Declare Function WSAStartup Lib "wsock32.dll" (ByVal wVersionRequired As Long, lpWSAData
As WSAData) As Long

Public Declare Function WSACleanup Lib "wsock32.dll" () As Long

Public Declare Function WSAUnhookBlockingHook Lib "wsock32.dll" () As Long

Public Declare Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" (hpvDest As Any, hpvSource As
Any, ByVal cbCopy As Long)
```

The basic control flow with WinSock API is shown in the figure below:

Control flow with WinSock API



e9071c372

The procedures of each step in Control flow with WinSock API are described below.

Startup

The procedure corresponding to Startup is StartIt. StartIt launches and initializes WinSock API with **WSAStartup**, whose version is shown in part 1 of ctrl_ lan.xls. The function WSAStartup should always be used when initiating WinSock. This function takes the version number (input) and launching information (output) as its parameters.

StartIt

```
Sub StartIt()

Dim StartUpInfo As WSAData
'Version 1.1 (1*256 + 1) = 257
```

```
'version 2.0 (2*256 + 0) = 512
'Get WinSock version
Sheets("Sheet1").Select
Range("C2").Select
version = ActiveCell.FormulaR1C1
'Initialize Winsock DLL
x = WSAStartup(version, StartUpInfo)

End Sub
```

Socket Creation and Connection

The procedure for Socket Creation and Connection is OpenSocket. OpenSocket makes a connection to an instrument associated with the IP address specified with the input parameter Hostname. It uses a socket of the port specified with the input parameter PortNumber. Each functional part of OpenSocket is described below.

In (1), the inet_aadr function of WinSock API is used to convert an IP address delimited by "." to an Internet address.

In (2), a new socket is created with the **socket** function of WinSock API and its socket descriptor is obtained. If an error occurs, the control returns to the main program with a message. The socket function takes an address family (input), a socket type (input), and a protocol number (input) as its parameters.

In (3), the socket address is specified. Note that htons, which is used for specifying the port number, is a function of WinSock API. This function converts a 2-byte integer from the Windows byte order (little endian) to the network byte order (big endian).

In (4), a connection to the E5061B is made by using the **connect** function of WinSock API. If an error occurs, the control returns to the main program with a message. The connect function takes a socket descriptor (input), a socket address (input), and the size of the socket address (input) as its parameters.

OpenSocket

```
Function OpenSocket(ByVal Hostname As String, ByVal PortNumber As Intege r) As Integer
Dim I_SocketAddress As sockaddr_in
Dim ipAddress As Long
ipAddress = inet_addr(Hostname) '.....(1)
```

```
'Create a new socket
socketId = socket(AF_INET, SOCK_STREAM, 0) '
If socketId = SOCKET_ERROR Then '
MsgBox ("ERROR: socket = " + Str$(socketId)) '.....(2)
OpenSocket = COMMAND_ERROR '
Exit Function '
End If '

'Open a connection to a server

I_SocketAddress.sin_family = AF_INET '
I_SocketAddress.sin_port = htons(PortNumber) '.....(3)
I_SocketAddress.sin_addr = ipAddress '
I_SocketAddress.sin_zero = String$(8, 0) '

x = connect(socketId, I_SocketAddress, Len(I_SocketAddress)) '
If socketId = SOCKET_ERROR Then '
MsgBox ("ERROR: connect = " + Str$(x)) '..(4)
OpenSocket = COMMAND_ERROR '
Exit Function '
End If '

OpenSocket = socketId

End Function
```

Communication

The procedure corresponding to Communication is SendCommand. SendCommand transmits a message (SCPI command) specified with the input parameter "command" to the E5061B using the **send** function of WinSock API. The send function takes a socket descriptor (input), a message to be transmitted (input), message length (input) and a flag (input) as its parameters.

SendCommand

```
Function SendCommand(ByVal command As String) As Integer

Dim strSend As String
strSend = command + vbCrLf
```

```
count = send(socketId, ByVal strSend, Len(strSend), 0)
If count = SOCKET_ERROR Then
MsgBox ("ERROR: send = " + Str$(count))
SendCommand = COMMAND_ERROR
Exit Function
End If
SendCommand = NO_ERROR

End Function
```

The procedure corresponding to the Receiving part of communication is RecvAscii and other functions. RecvAscii receives a message in ASCII format and stores it in the dataBuf output parameter. Maximum length of the message is specified with the maxLength input parameter. Each functional part of RecvAscii is described below.

In (1), a message (a response to a query for a SCPI command) is received from the E5061B as a series of characters using the **recv** function of WinSock API. If an error occurs, the control returns to the main program with a message. The recv function takes a socket descriptor (input), a message to be received (input), message length (input) and a flag (input) as its parameters.

In (2), it is determined whether each received character is LF (ASCII code: 10). When it is LF, receiving is terminated by adding NULL (ASCII code: 0) to the end of the dataBuf string and the control returns to the main program.

In (3), the number of the last characters that were read out is added to the count value for checking the number of received characters, and the characters are appended to the end of the dataBuf string.

RecvAscii

```
Function RecvAscii(dataBuf As String, ByVal maxLength As Integer) As Integer

Dim c As String * 1
Dim length As Integer
dataBuf = ""
While length < maxLength
DoEvents
count = recv(socketId, c, 1, 0) '
If count < 1 Then '
RecvAscii = RECV_ERROR '.....(1)
dataBuf = Chr$(0) '
```

```
Exit Function '
End If '
If c = Chr$(10) Then '
dataBuf = dataBuf + Chr$(0) '.....(2)
RecvAscii = NO_ERROR '
Exit Function '
End If '
length = length + count '.....(3)
dataBuf = dataBuf + c '
Wend
RecvAscii = RECV_ERROR
End Function
```

Disconnection

The procedure corresponding to Disconnection is CloseConnection. CloseConnection disconnects communication and removes a socket using the **closesocket** function of WinSock API. The closesocket function takes a socket descriptor (input) as its parameter.

CloseConnection

```
Sub CloseConnection()

x = closesocket(socketId)
If x = SOCKET_ERROR Then
MsgBox ("ERROR: closesocket = " + Str$(x))
Exit Sub
End If

End Sub
End
```

The procedure corresponding to End is EndIt. EndIt disconnects WinSock API using the **WSACleanup** function of WinSock API. The function WSACleanup should always be used when terminating WinSock.

EndIt

```
Sub EndIt()

'Shutdown Winsock DLL
```

```
x = WSACleanup()

End Sub
```

Example of control

The E5061B can be controlled by executing the above procedures in order, following the control flow in Control flow with WinSock API. This is demonstrated by the procedure autoscale (a procedure that is executed when the Auto Scale button is clicked) as described in autoscale.

autoscale

```
Sub autoscale()
'
' auto scaling
'
Call StartIt
Call get_hostname
x = OpenSocket(Hostname$, ScpiPort)
x = SendCommand(":DISP:WIND1:TRAC1:Y:AUTO")
Call CloseConnection
Call EndIt
End Sub
```

When you execute more than one command by connecting and disconnecting a socket for every command, the sequence of execution may change.

Control LCD Update Timing

- Overview
- Sample Program in Excel VBA using VISA
- Sample Program in HT Basic

Other topics about Sample Programs

Overview

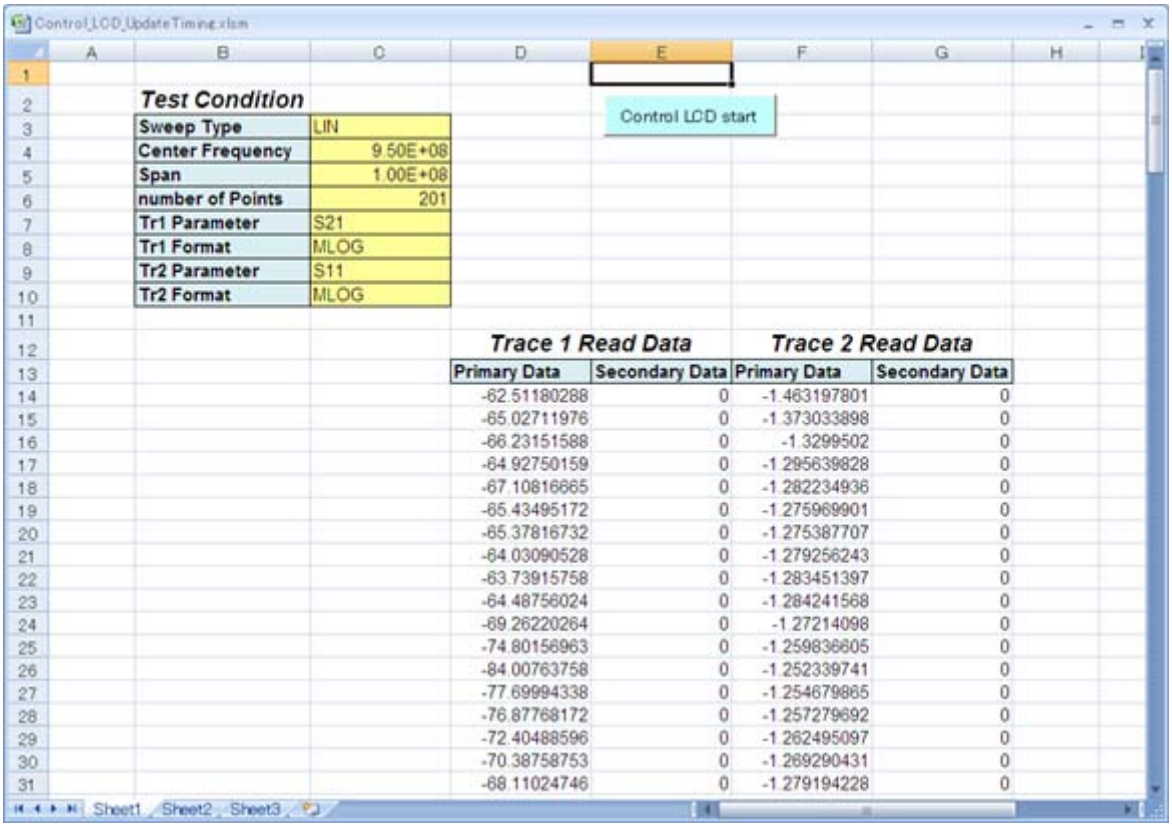
This sample program is provided in this section where the command processing time is improved by controlling the update timing of the LCD display.

NOTE This sample program correctly runs when the maximum number of channels/traces is set to 4 channels/4 traces.

This program sets the necessary measurement conditions and then turns OFF the updating of the LCD display. Next, it performs measurement, reads out the result, and updates the screen once. This program repeats this measurement procedure ten times.

Sample Program in Excel VBA using VISA

Example of excel sheet with control LCD update timing program



```
Private Sub Ctrl_LCD_Click()
    Dim defrm As Long
    Dim Age506x As Long

    Dim SwType As String
    Dim Cent As Double, Span As Double
    Dim Param(1) As String, Fmt(1) As String
    Dim NumPoin As Integer, NumData As Integer

    Const SwTime = 2#

    Dim Dummy As String * 20

    Dim Tr1ptr(3) As Long
    Dim Tr1Data() As Double
    Dim Tr2ptr(3) As Long
    Dim Tr2Data() As Double

    Dim WrtFmt As String

    '***
    '*** Open session.
    '***

    Call viOpenDefaultRM(defrm)
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, Age506x)
    Call viSetAttribute(Age506x, VI_ATTR_TMO_VALUE, 10000)

    '***
    '*** Set variable of measurement condition.
    '***

    SwType = Trim(Cells(3, 3).Value)
    Cent = CDbl(Cells(4, 3).Value)
    Span = CDbl(Cells(5, 3).Value)
    NumPoin = CInt(Cells(6, 3).Value)
```

```
Param(0) = Trim(Cells(7, 3).Value)
Fmt(0) = Trim(Cells(8, 3).Value)
Param(1) = Trim(Cells(9, 3).Value)
Fmt(1) = Trim(Cells(10, 3).Value)

'***
'*** Send measurement condition to E5061B.
'***

Call viVPrintf(Age506x, ":SENS1:SWE:TYPE " + Trim(SwType) + vbLf, 0)
Call viVPrintf(Age506x, ":SENS1:FREQ:CENT " + CStr(Cent) + vbLf, 0)
Call viVPrintf(Age506x, ":SENS1:FREQ:SPAN " + CStr(Span) + vbLf, 0)
Call viVPrintf(Age506x, ":SENS1:SWE:POIN " + CStr(NumPoin) + vbLf, 0)

Call viVPrintf(Age506x, ":TRIG:SOUR BUS" + vbLf, 0)
Call viVPrintf(Age506x, ":INIT1:CONT ON" + vbLf, 0)
Call viVPrintf(Age506x, ":SENS1:SWE:TIME:AUTO OFF" + vbLf, 0)
Call viVPrintf(Age506x, ":SENS1:SWE:TIME " + CStr(SwTime) + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" + vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)

For i = 2 To 4
   Call viVPrintf(Age506x, ":INIT" + CStr(i) + ":CONT OFF" + vbLf, 0)
Next i

Call viVPrintf(Age506x, ":DISP:SPL D1" + vbLf, 0)
Call viVPrintf(Age506x, ":DISP:WIND1:SPL D1_2" + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" + vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)

'***
'*** Setting Trace 1 and 2.
'***

Call viVPrintf(Age506x, ":CALC1:PAR:COUN 2" + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:PAR1:DEF " + Trim(Param(0)) + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:PAR1:SEL" + vbLf, 0)
```

```
Call viVPrintf(Age506x, ":CALC1:FORM " + Trim(Fmt(0)) + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:PAR2:DEF " + Trim(Param(1)) + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:PAR2:SEL" + vbLf, 0)
Call viVPrintf(Age506x, ":CALC1:FORM " + Trim(Fmt(1)) + vbLf, 0)

Call viVPrintf(Age506x, ":DISP:ENAB OFF" + vbLf, 0)
Call viVPrintf(Age506x, ":FORM:DATA REAL" + vbLf, 0)
Call viVPrintf(Age506x, "*OPC?" + vbLf, 0)
Call viVScanf(Age506x, "%t", Dummy)

'***
'*** redim for read data.
'***

NumData = NumPoin * 2

ReDim Tr1Data(NumData - 1)
Tr1ptr(0) = VarPtr(NumData)
Tr1ptr(1) = VarPtr(Tr1Data(0))

ReDim Tr2Data(NumData - 1)
Tr2ptr(0) = VarPtr(NumData)
Tr2ptr(1) = VarPtr(Tr2Data(0))
WrtFmt = "%#Zb%1t"

'***
'*** Cycling trigger, read data, and screen update.
'***

For i = 1 To 10

    '***
    '*** Trigger.
    '***

    Call viVPrintf(Age506x, ":TRIG:SING" + vbLf, 0)
    Call viVPrintf(Age506x, "*OPC?" + vbLf, 0)
```

```
        Call viVScanf(Age506x, "%t", Dummy)

        '***
        '*** Read trace data.
        '***

        Call viVPrintf(Age506x, ":CALC1:PAR1:SEL" + vbLf, 0)
        Call viVPrintf(Age506x, ":CALC1:DATA:FDAT?" + vbLf, 0)
        Call viVScanf(Age506x, WrtFmt, Tr1ptr(0))

        For j = 0 To NumData / 2 - 1
            Cells(14 + j, 4).Value = Tr1Data(j * 2)
            Cells(14 + j, 5).Value = Tr1Data(j * 2 + 1)
        Next j

        Call viVPrintf(Age506x, ":CALC1:PAR2:SEL" + vbLf, 0)
        Call viVPrintf(Age506x, ":CALC1:DATA:FDAT?" + vbLf, 0)
        Call viVScanf(Age506x, WrtFmt, Tr2ptr(0))

        For j = 0 To NumData / 2 - 1
            Cells(14 + j, 6).Value = Tr2Data(j * 2)
            Cells(14 + j, 7).Value = Tr2Data(j * 2 + 1)
        Next j

        '***
        '*** screen update.
        '***

        Call viVPrintf(Age506x, ":DISP:UPD" + vbLf, 0)
    Next i

    Call viVPrintf(Age506x, ":FORM:DATA ASC" + vbLf, 0)
    Call viClose(Age506x)
    Call viClose(defrm)

End Sub
```

Sample Program in HT Basic (cont_upd.htb)

```
10 REAL Trace1(1:201,1:2),Trace2(1:201,1:2)
20 DIM Buff$[9],Img$[30]
30 INTEGER Nop,I
40 !
50 ASSIGN @Agte506x TO 717
60 ASSIGN @Binary TO 717;FORMAT OFF
70 !
80 OUTPUT @Agte506x;":SENS1:SWE:TYPE LIN"
90 OUTPUT @Agte506x;":SENS1:FREQ:CENT 950E6"
100 OUTPUT @Agte506x;":SENS1:FREQ:SPAN 100E6"
110 OUTPUT @Agte506x;":SENS1:SWE:POIN 201"
120 OUTPUT @Agte506x;":TRIG:SOUR BUS"
130 OUTPUT @Agte506x;":INIT1:CONT ON"
140 FOR I=2 TO 4
150 OUTPUT @Agte506x;":INIT"&VAL$(I)&":CONT OFF"
160 NEXT I
170 !
180 OUTPUT @Agte506x;":DISP:SPL D1"
190 OUTPUT @Agte506x;":DISP:WIND1:SPL D1_2"
200 !
210 OUTPUT @Agte506x;":CALC1:PAR:COUN 2"
220 OUTPUT @Agte506x;":CALC1:PAR1:DEF S21"
230 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
240 OUTPUT @Agte506x;":CALC1:FORM MLOG"
250 OUTPUT @Agte506x;":CALC1:PAR2:DEF S11"
260 OUTPUT @Agte506x;":CALC1:PAR2:SEL"
270 OUTPUT @Agte506x;":CALC1:FORM MLOG"
280 !
290 OUTPUT @Agte506x;":DISP:ENAB OFF"
300 OUTPUT @Agte506x;":FORM:DATA REAL"
310 !
320 FOR I=1 TO 10
330 OUTPUT @Agte506x;":TRIG:SING"
340 OUTPUT @Agte506x;"*OPC?"
350 ENTER @Agte506x;Buff$
360 !
```

```
370 ! Read Trace Data
380 !
390 OUTPUT @Agte506x;":CALC1:PAR1:SEL"
400 OUTPUT @Agte506x;":CALC1:DATA:FDAT?"
410 ENTER @Agte506x USING "#,8A";Buff$
420 ENTER @Binary;Trace1(*)
430 ENTER @Agte506x USING "#,1A";Buff$
440 !
450 OUTPUT @Agte506x;":CALC1:PAR2:SEL"
460 OUTPUT @Agte506x;":CALC1:DATA:FDAT?"
470 ENTER @Agte506x USING "#,8A";Buff$
480 ENTER @Binary;Trace2(*)
490 ENTER @Agte506x USING "#,1A";Buff$
500 !
510 ! Update Display
520 !
530 OUTPUT @Agte506x;":DISP:UPD"
540 NEXT I
550 END
```

Description

Lines 50 to 60

Assigns a GPIB address to the I/O pass.

Lines 80 to 110

These lines set the sweep type to linear sweep, the sweep center value to 950 MHz, the sweep span value to 100 MHz, and the number of measurement points to 201.

Lines 120 to 160

These lines set the trigger source to bus trigger, turn ON Continuous Activation mode for channel 1, and turn the mode OFF for channels 2 through 4.

Lines 180 to 190

These lines display the window for channel 1 only and arrange two graphs tiled horizontally.

Lines 210 to 270

These lines set the number of traces for channel 1 to 2, the measurement parameter and its data format for trace 1 to S21 and Log Mag, respectively, and those for trace 2 to S11 and Log Mag, respectively.

Line 290

This line turns OFF the updating of the LCD screen.

Line 300

This line sets the data transfer format to binary.

Lines 320 to 540

These lines repeat the following procedure ten times.

Lines 340 to 360: These lines trigger the instrument and wait until the measurement cycle finishes.

Lines 400 to 440: Reads out the formatted data array of trace 1 in channel 1.

Lines 460 to 500: Reads out the formatted data array of trace 2 in channel 1.

Line 540: This line updates the LCD screen once.

Handler Interface

- Overview
- Program Code

Other topics about Sample Programs

Overview

The sample program communicates with an external instrument through the handler I/O port.

This program outputs 5 (sets bit 2 and bit 0 to Low, and the other bits to High) to the port A of the handler I/O port, then waits until the bit 3 of the port C is set to Low.

See Inputting/Outputting Data for this programming.

Program Code

Excel VBA

```
Sub Handler_Click()
    Dim defrm As Long          'Session to Default Resource Manager.
    Dim vi As Long             'Session to instrument.
    Dim Out_Data As Integer    'Decimal value.
    Dim In_Data As Long
    Dim Bit_stat As Integer
    Dim Flag_bit As Integer
    Dim Out_Data_Bin As String
    Dim Ret As Long            'Return value.
    Dim i As Long
    Dim X As Long
    Const TimeOutTime = 40000    'timeout time.
    Out_Data_Bin = "00000101"    'Store the output data on the port A (binaly).
    Flag_bit = 3                 'Bit location (bit 3).

    Call viOpenDefaultRM(defrm)    'Initializes the VISA system.
    Call viOpen(defrm, "GPIB0::17::INSTR", 0, 0, vi)   'Opens the session to the specified instrument.
    Call viSetAttribute(vi, VI_ATTR_TMO_VALUE, TimeOutTime)   'The state of an attribute for the
specified session.

    Call viVPrintf(vi, "*RST" & vbLf, 0)    'Presets the setting state of the ENA-L.
    Call viVPrintf(vi, "*CLS" & vbLf, 0)   'Clears the all status register.
```

```
    Call viVPrintf(vi, ":CONT:HAND:C:MODE INP" & vbLf, 0)     'Configures the port C to input port.
    Call viVPrintf(vi, ":CONT:HAND:IND:STAT ON" & vbLf, 0)    'Line enable /INDEX signal.
    Call viVPrintf(vi, ":CONT:HAND:RTR:STAT ON" & vbLf, 0)   'Line enable /READY FOR TRIGGER
signal.

    For i = 1 To Len(Out_Data_Bin)    'Convert Out_Dara_Bin to a decimal value.
        If Mid(Out_Data_Bin, Len(Out_Data_Bin) - i + 1, 1) = "1" Then
            X = 2 ^ (i - 1)
            Ret = Ret + X
        End If
    Next i
    Out_Data = Ret       'Sets the decimal value.
    Call viVPrintf(vi, ":CONT:HAND:A " & Ret & vbLf, 0)    'Sets to the port A.

    Call viVPrintf(vi, ":CONT:HAND:C?" & vbLf, 0)   'Outputs data to output port C.
    Call viVScanf(vi, "%t", In_Data)    'Reads data from the port C.

    Call ErrorCheck(vi)      'Checking the error.

    Call viClose(vi)          'Closes the resource manager session.
    Call viClose(defrm)     'Breaks the communication and terminates the VISA system.

    End
End Sub
Sub ErrorCheck(vi As Long)
    Dim err As String * 50, ErrNo As Variant, Response

    Call viVQueryf(vi, ":SYST:ERR?" & vbLf, "%t", err)    'Reads error message.
    ErrNo = Split(err, ",")  'Gets the error code.

    If Val(ErrNo(0)) <> 0 Then
        Response = MsgBox(CStr(ErrNo(1)), vbOKOnly)  'Display the message box.
    End If
End Sub
```

HT Basic (handler.htb)

```
10 INTEGER Out_data,In_data,Bit_stat
20 DIM Out_data_bin$[9]
```

```
30 !
40 ASSIGN @Agte506x TO 717
50 !
60 Out_data_bin$="00000101"
70 Flag_bit=3
80 !
90 OUTPUT @Agte506x;":CONT:HAND:C:MODE INP"
100 OUTPUT @Agte506x;":CONT:HAND:IND:STAT ON"
110 OUTPUT @Agte506x;":CONT:HAND:RTR:STAT ON"
120 !
130 Out_data=IVAL(Out_data_bin$,2)
140 OUTPUT @Agte506x;":CONT:HAND:A ";Out_data
150 !
160 REPEAT
170 OUTPUT @Agte506x;":CONT:HAND:C?"
180 ENTER @Agte506x;In_data
190 Bit_stat=BIT(In_data,Flag_bit)
200 UNTIL Bit_stat=1
210 END
```

VBA Programming

VBA Programming (Embedded VBA)

- Introduction to VBA Programming
- Operation Basics
- Controlling E5061B
- Controlling Peripherals
- Application Programs
- Complex Operation Library
- Waveform Analysis Library

Introduction to VBA Programming

Introduction to VBA Programming

- Introduction of the E5061B Macro Function
- An Overview of a Control System Based on the Macro Function
- Overview of E5061B COM Object

Introduction of the E5061B Macro Function

Other topics about Introduction to VBA Programming

Overview

The E5061B has a built-in macro function that allows a single instruction to substitute for multiple instructions. You can have the E5061B to automatically execute your own macro program that contains a series of VBA (Visual Basic for Application) statements. The macro function allows you to run a variety of applications; you can control not only the E5061B but also various peripherals from your own macro code.

The VBA is based on the VB (Visual Basic) programming language. Although the VBA is similar to the VB, they are not the same. The VBA has decreased some of the VB's features and added characteristic features to each application. The E5061B VBA is an added feature for controlling the E5061B. For details on the difference between the VBA and the VB, refer to Microsoft official guides, and various books on VBA.

For information on the basic operating procedures for the E5061B's VBA, see Operation Basics of the E5061B's VBA. This manual is not meant to be an in-depth guide to VBA programming basics and the syntax of VBA functions and commands. Such in-depth information is covered in VBA Help, Microsoft official guides, and various books on VBA.

Macro Function

The macro function allows you to control the E5061B itself as well as various peripherals. You can do the following:

1. Automate repetitive tasks
   You can use the E5061B's macro function to combine several processes into one. Automating repetitive tasks provides higher efficiency and eliminates human error. Once you have contained repetitive tasks in Sub procedures, you can later call the procedures from other programs, thus allowing effective reuse of programming assets.
2. Implement a user interface
   The E5061B VBA supports user forms that simplify visual user interface creation. User forms guide users through common tasks such as performing measurement and entering data, without requiring familiarity with the E5061B, thus minimizing the possibility of human error.

An Overview of a Control System Based on the Macro Function

Other topics about Introduction to VBA Programming

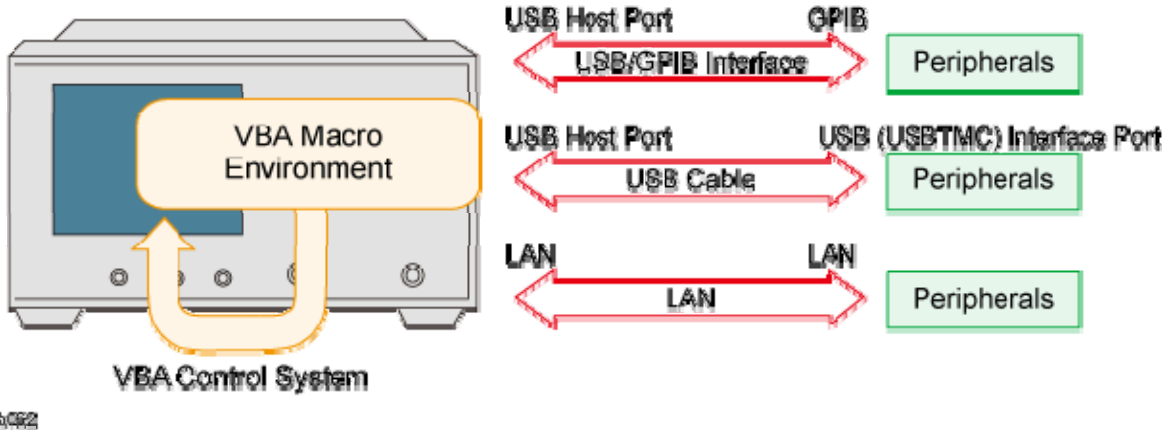
Overview

This section describes how you can use the E5061B's built-in VBA macro function to implement a system that controls the E5061B and peripherals, and what command sets are available for such purposes.

Implementing a Control System

Macro-based control systems are classified into two types: As shown in the following figure, a VBA control system controls the E5061B itself while a VBA remote control system controls peripherals. When you use the macro function to control the peripherals, you must connect the E5061B with the peripherals through USB/GPIB interface, USB or LAN, and configure them to communicate over VISA (Virtual Instrument Software Architecture). For information on programming using the VISA library, refer to Programming with VISA.

Configuration example of control system using macro environment



Required Equipment

- E5061B
- Peripherals and/or other instruments that serve your purpose
- USB/GPIB interface, USB Cable, or LAN

Control Methods

The command set you can use differs depending on whether you use the macro function to control the E5061B or a peripheral.

Controlling the E5061B

When you want to control the E5061B itself, you can create a program using COM objects within the E5061B VBA environment. COM objects that come with the E5061B include seven objects specific to the COM interface and COM objects that correspond to SCPI commands.

Controlling a Peripheral

When you want to control a peripheral, you can create a program using VISA library functions within the E5061B VBA environment.

For information on using the VISA library, see Controlling Peripherals. For a complete description on VISA functions, refer to the VISA library's online help.

For information on the GPIB commands that is available with a particular peripheral, refer to the documentation that comes with the peripheral.

Overview of E5061B COM Object

Other topics about Introduction to VBA Programming

Overview

The E5061B VBA environment provides COM objects that support the E5061B control. This section provides an overview of COM objects as well as considerations for using the E5061B's COM objects.

The definitions and specifications of COM are beyond the scope of this guide. Such in-depth information is covered in various books on COM.

About COM Object

When you control the E5061B through the macro function, you can use COM objects as components of your application. The functionality of the E5061B's COM objects is exposed through properties and methods.

Property

A property allows you to read or write a setting or attribute of an object. With the E5061B, you can use properties to set or read the settings of the E5061B.

You can find properties in the list of object types in COM Object Reference.

Method

A method allows you to manipulate an object in a particular way. With the E5061B, you can use methods to perform specific tasks.

You can find methods in the list of object types in COM Object Reference.

Event

An event means an operation from outside that the program can recognize such as clicking a mouse. The E5061B detects events when a specific softkey is pressed using the UserMenu_OnPress(ByVal Key_id As Long) procedure to execute the assigned procedure.

You can find events in the list of object types in COM Object Reference.

Using COM Object to Control E5061B

When you want to control the E5061B, you can use COM objects alone or in conjunction with SCPI commands and the Parse object. The latter method is a little slower than the former method because the Parse object

is used to parse the messages of SCPI commands. For instructions on using the E5061B's VBA Editor to create a program that uses COM objects, refer to Operation Basics of the E5061B's VBA.

Major Control Difference between COM Object and SCPI Command

While the control using SCPI commands allows SRQ (Service Request) interruptions through the status reporting mechanism, the control using COM objects does not support SRQ interruptions. Instead of SRQ interrupts, you can use the WaitOnSRQ object to suspend the program until the E5061B is placed into the desired state.

Operation Basics

Operation Basics

- Displaying Visual Basic Editor
- Closing Visual Basic Editor
- Switching to the E5061B Measurement Screen
- Making a Preparation Before Coding
- Coding a VBA Program
- Saving a VBA program
- Loading a VBA Program
- Running a VBA Program
- Stopping a VBA Program
- Errors and Debugging
- Printing Output Values in the Echo Window
- Uses Advanced Techniques
- Using VBA Online Help

Displaying Visual Basic Editor

Other topics about Operation Basics

Overview

This section describes how to launch Visual Basic Editor.

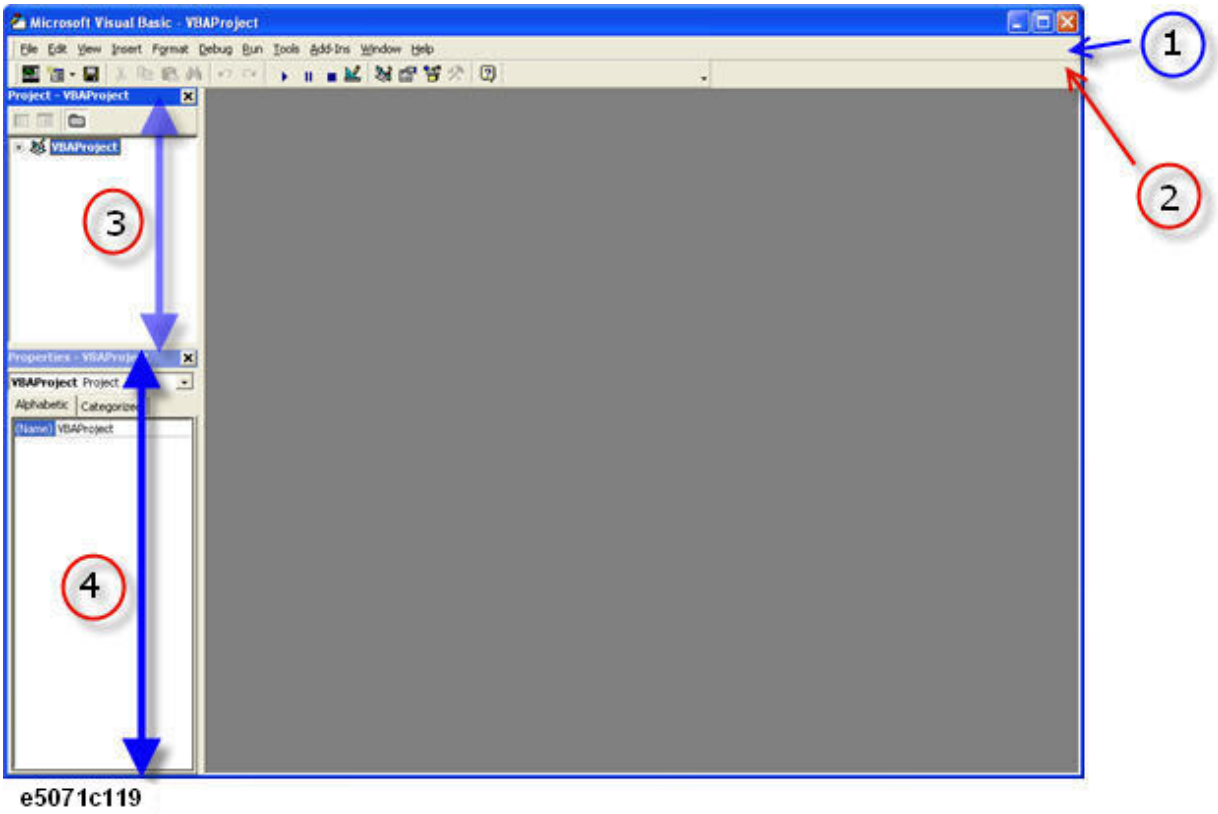
From the E5061B measurement screen, launch Visual Basic Editor using one of the following methods:

1. **Macro Setup** > **VBA Editor**
2. Press **Alt** + **F11** keys on the keyboard.

Initial Screen of Visual Basic Editor

When you launch Visual Basic Editor, it displays the initial screen, which contains a number of windows as shown in the following figure. The initial screen provides the following GUI elements:

Example of Visual Basic Editor initial screen



e5071c119

1. Menu bar
2. Toolbar
3. Project Explorer

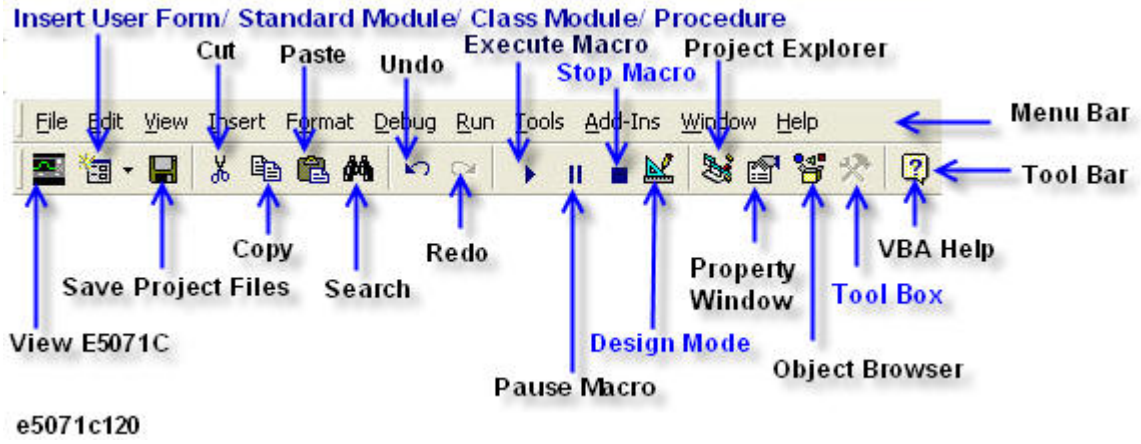
4. Property Window

Menu Bar

Clicking one of the menu labels brings up the corresponding menu. The menu bar can be used as the primary method to navigate through E5061B's VBA environment.

Toolbar

The toolbar provides access to commonly used commands via icon buttons; these commands are a subset of the commands accessible from the menu bar.



Project Explorer

Within the E5061B's VBA environment, you can develop your application as a project that consists of a number of files (modules). Project Explorer shows a list of all files (modules) that make up a project. The list also includes files (modules) created or loaded in Visual Basic Editor. For information on modules, refer to A Project and Three Types of Module.

To display the project explorer, do one of the following:

1. On the **View** menu, click **Project Explorer**.
2. Press **Ctrl** + **R** keys on the keyboard.
3. On the toolbar, click  .

Property Window

A property window shows the settings (label, font, color, size, etc.) of a control (such as a command button or text box) placed on the user form. For information on user forms, refer to User Form.

You can also set properties by programming in the code window.

To display the properties window, do one of the following:

1. On the **View** menu, click **Properties Window**.
2. Press **F4** key on the keyboard.
3. On the toolbar, click  .

Closing Visual Basic Editor

This section describes how to quit Visual Basic Editor. Close the Visual Basic Editor using either one of the following methods:

- On Visual Basic Editor's **File** menu, click **Close and Return to E5061**.
- Within Visual Basic Editor, press **Alt** + **Q** keys on the keyboard.
- **Macro Setup** > **Close Editor** (E5061B measurement screen)

NOTE Whenever you launch Visual Basic Editor, it automatically displays the project files you were working with in the previous session. However, once you turn OFF the power to the E5061B, the project files kept in memory will be lost; therefore, it is strongly recommended to save your VBA programs before you turn OFF the power.

Other topics about Operation Basics

Switching to the E5061B Measurement Screen

You can switch to the E5061B measurement screen without closing Visual Basic Editor.

- On the **View** menu, click **E5061**.
- Press **Alt** + **F11** keys on the keyboard.
- On the toolbar, click "E5061B" icon.
- Press **Foc** key on the E5061B front panel.

Other topics about Operation Basics

Making a Preparation Before Coding

- A Project and Three Type of Modules
- Displaying a Code Window

Other topics about Operation Basics

A Project and Three Type of Modules

Project Explorer displays a list of files (modules) that are used in the E5061B VBA. This section describes a project composed of a number of files (modules) and three types of modules ("user form", "standard," and "class"). Each type of module serves its own purpose as described below.

Project

When you develop an application within the E5061B's VBA environment, you use a number of VBA program files (modules), and manage them as one project. The project is saved with the file extension ".vba".

User Form

A user form contains controls such as buttons and text boxes. You can code event-driven procedures that are invoked when a particular event occurs on a particular control, thereby creating a user interface. The user form is saved with the file extension ".frm".

Standard module

A standard module contains a collection of one or more procedures (subprograms enclosed between Sub and End Sub). One typical use of a standard module is to contain shared subroutines and globally called functions. The standard module is saved with the file extension ".bas".

Class Module

A class module contains both data and procedures and acts as one object. Once you have created a class module that serves as an object, you can create any number of instances of that object by naming each instance as an object variable. While each procedure must be unique in a standard module, you can have multiple instances of an object created through a class module. The class module is saved with the file extension ".cls".

Displaying a Code Window

The code window appear on the Visual Basic Editor by inserting the modules in a project. You can practically do coding (programming) on this code window.

The E5061B's VBA environment does not allow you to manage multiple projects. When the current project is exists in the Visual Basic Editor by loading the saved project file, you can replace the current project with a new project by the following method shown the E5061B measurement screen.

1. **Macro Setup** > **New Project**

   NOTE When you replace the current project with a new project, a message prompts to save the current project. If you want to save the project, click **Yes** button to display a dialog box for saving. For saving the project, see Saving a Project.

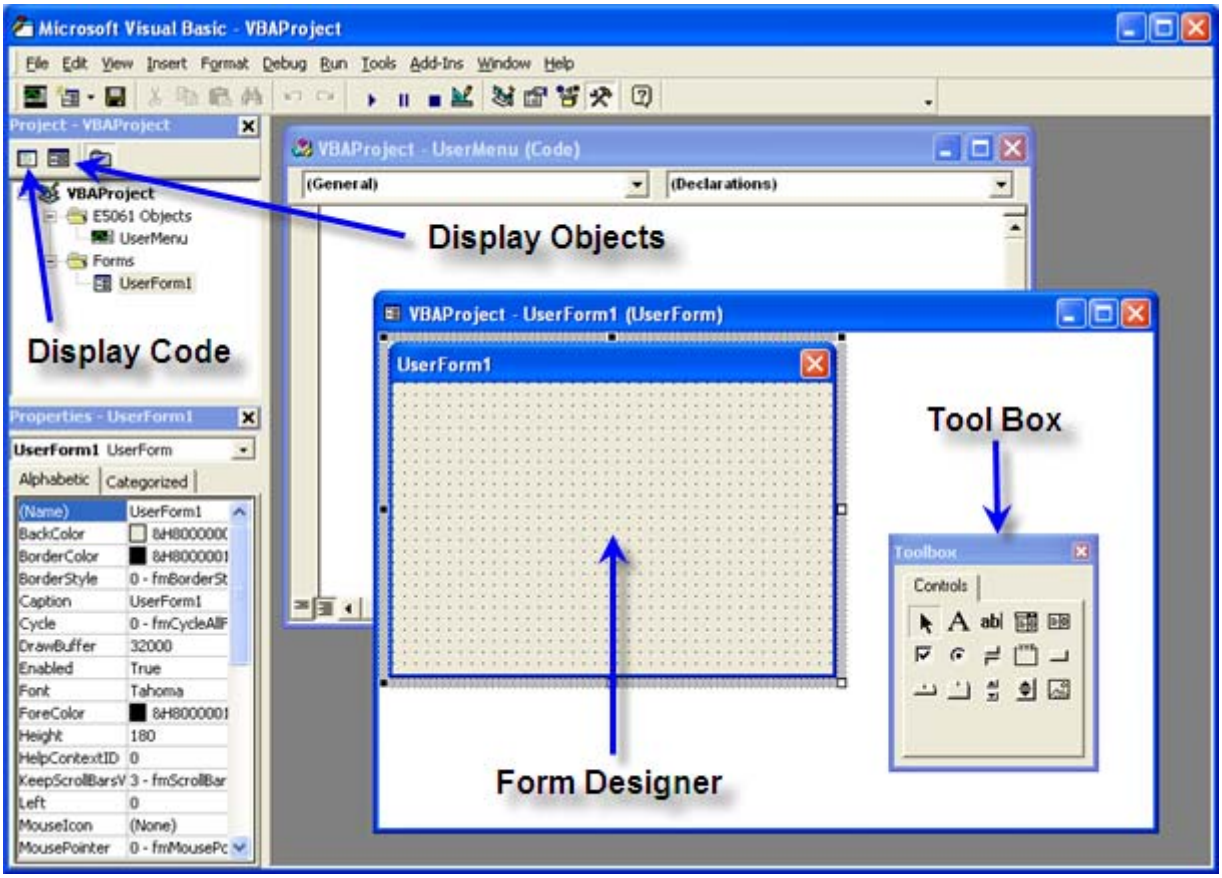
Inserting the User Form

Within Visual Basic Editor, do one of the following to add a user form to your project.

1. On the **Insert** menu, click **UserForm**.
2. On the toolbar, click "Insert User Form/Standard Module/Class Module/Procedure" icon, and click **UserForm**.
3. In Project Explorer, right-click the "VBAProject" icon, and click **Insert** > **UserForm**.

   NOTE Adding a user form does not automatically open the code window for that user form. To open the code window, click the "Display Code" icon on Project Explorer in the following figure or double-click a control placed on the user form.

Adding a user form

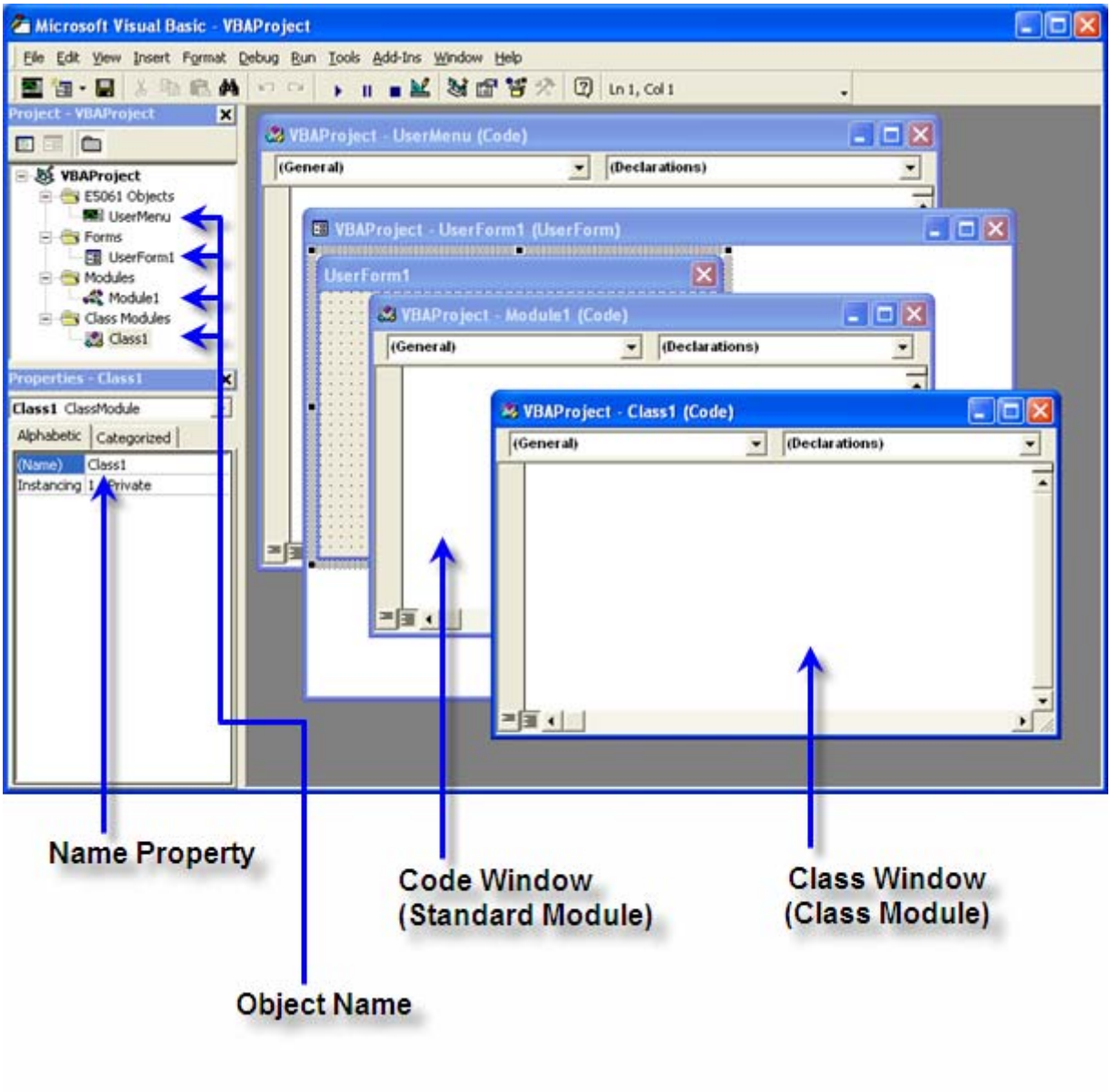


Inserting the Standard Module

Within Visual Basic Editor, do one of the following to add a standard module to your project.

1. On the **Insert** menu, click **Module**.
2. On the toolbar, click "Insert User Form/Standard Module/Class Module/Procedure" icon, and click **Module**.
3. In Project Explorer, right-click the "VBAProject" icon, and click **Insert** > **Module**.

Adding a standard module/class module



Inserting the Class Module

Within Visual Basic Editor, do one of the following to add a class module to your project.

1. On the **Insert** menu, click **ClassModule**.
2. On the toolbar, click "Insert User Form/Standard Module/Class Module/Procedure" icon, and click **ClassModule**.
3. In Project Explorer, right-click the "VBAProject" icon, and click **Insert** > **ClassModule**.

Deleting Modules

You can delete any unnecessary module from the project within Visual Basic Editor. The following procedure assumes that you want to delete a class module named "Class1".

1. In Project Explorer, click the "Class1" class module under the "Class Modules" icon to highlight it.
2. Delete the "Class1" class module using one of the following methods:
    a. On the **File** menu, click **Remove Class1…**.
    b. Click the right mouse button, and click **Remove Class1…**.
3. When you are prompted to confirm whether to export (save) "Class1", click **No**. Alternatively, you can click **Yes** if you want to save the module.

Coding a VBA Program

- Overview
- User Interface Elements of a Code Window
- Creating a Simple VBA Program
- Auto-complete Feature

Other topics about Operation Basics

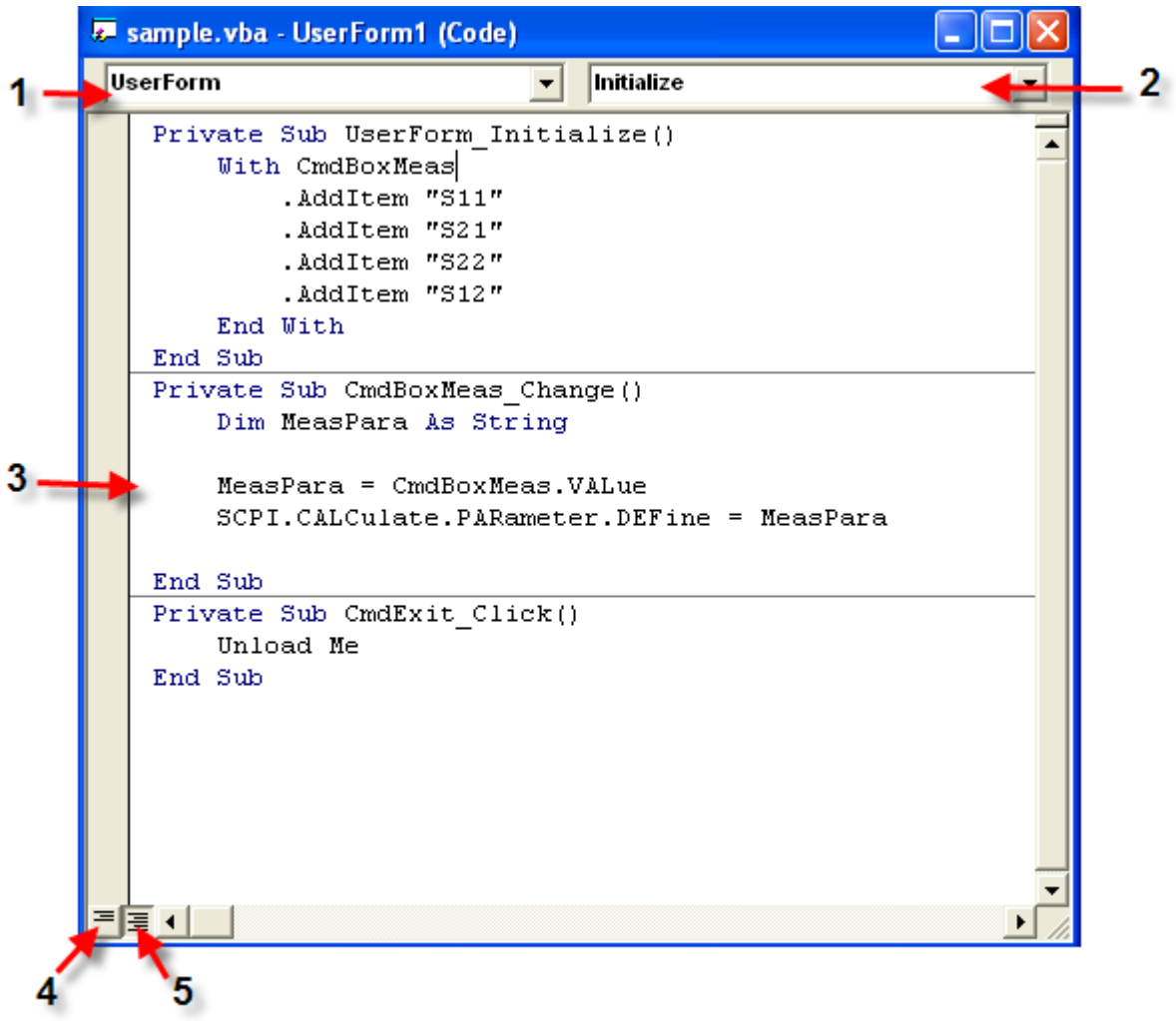
Overview

This section provides descriptive information on the user interface elements of a code window that lets you code a VBA program, and walks through a sample program (procedure) that finds the maximum value contained in an array so you can gain insight into how to create your own programs.

User Interface Elements of a Code Window

A code window is where you code a VBA program. When you are working with a user form, you can open the code window for that user form by double-clicking a control (such as a button or text box) placed on the form. Similarly, when you are working with a standard or class module, you can open the code window associated with that module by double-clicking the module's icon in Project Explorer.

Code window for a standard module



e5061b070

1. Object box

Provides a list of objects currently used within the code window.

2. Procedure box

Provides a list of procedures that reside within the code window. When you are working with a user form, this provides a list of events (actions such as click or double-click).

3. Margin indicator bar

Primarily intended for use when debugging a program.

4. Show Procedure button

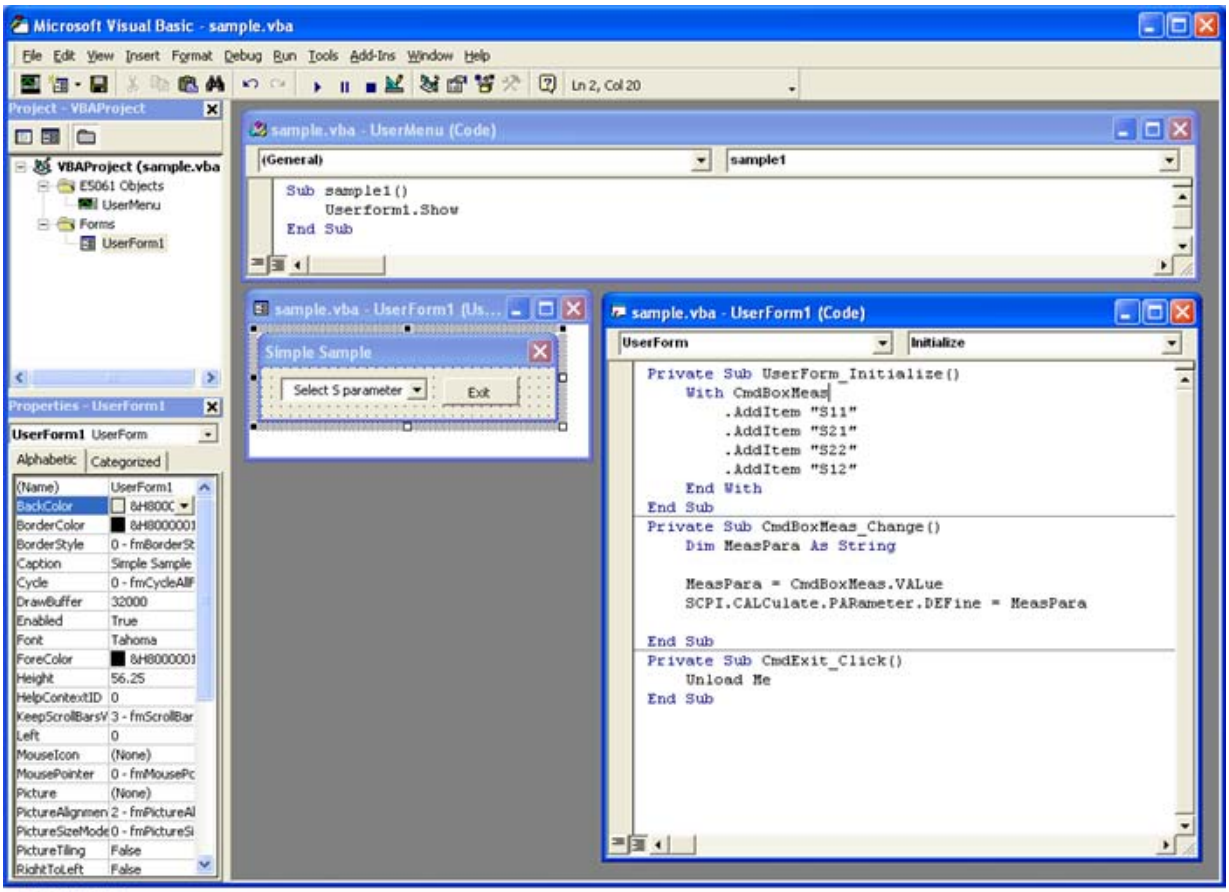
Displays only the procedure at the cursor position.

5. Show Module button

Displays the entire program contained in the code window.

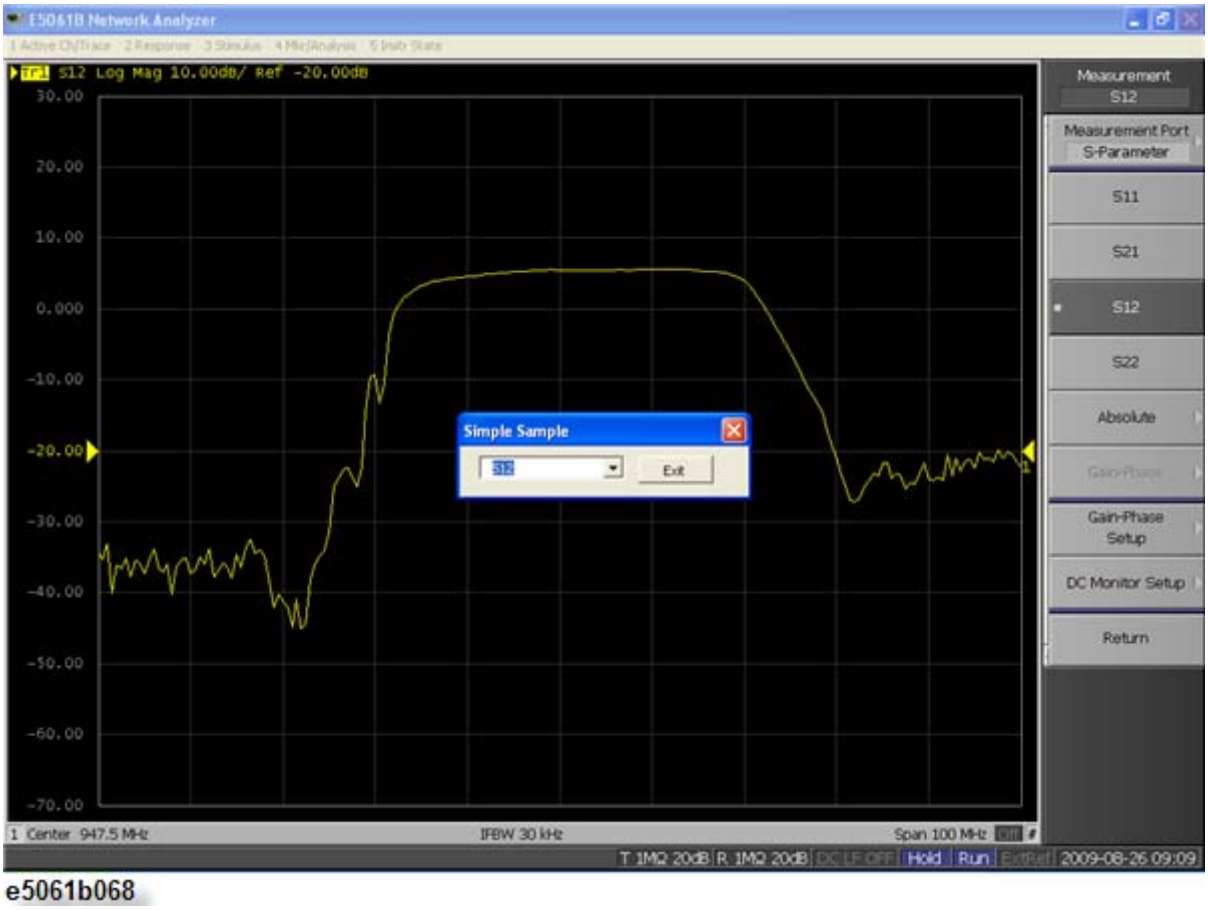
Creating a Simple VBA Program

The following figure shows a simple sample program. This program allows you to select the measurement parameter from S11, S21, S22 and S12.



e5061b069

When you run this program, the following dialog box is displayed and you can change measurement parameter by selecting parameter from combo box. Clicking the exit button quit the program.



Auto-complete Feature

When you use COM objects in Visual Basic Editor, the editor's auto-complete feature allows you to easily type in keywords without misspelling them.

The following procedure assumes that you are entering the SCPI.INITiate(Ch).CONTinuous object.

1. In a standard module, type **sub main** and press the **Enter** key. **End Sub** is automatically added.
2. Typing **scpi** followed by a dot (.) brings up a list of classes under the SCPI class.
3. Typing **in** automatically moves focus to **INITiate** in the list box.
4. Typing **(** brings up a list of indexes.
5. Typing **1).** brings up a list of classes under the INITiate class.
6. Typing **c** automatically moves focus to **CONTinuous** in the list box.
7. Typing **=** brings up a list box for setting a Boolean value (**True/False**).
8. Typing **t** automatically moves focus to **True**.

9. Pressing the **Enter** key completes the statement: SCPI.INITiate(1).CONTinuous = True.

Saving a VBA program

- Overview
- Saving a Project
- Saving a Module (Exporting)

Other topics about Operation Basics

Overview

You can save VBA programs either as one complete project or on a module by module basis.

Saving a Project

When you opt to save your program as one complete project, you can have the files (modules) making up the project into a single package. A project is saved as a .vba file. You can save your program to a project file using one of the following two methods:

Saving a Project from Visual Basic Editor

1. Open the Save As dialog box by doing one of the following:
   - On the **File** menu, click **Save xxx.VBA**. "**xxx**" represents the file name.
   - On the toolbar, click "Save Project File" icon.
   - Press **Ctrl** + **S** keys on the keyboard.
2. The Save As dialog box appears. Specify the file name and location (drive or folder) and click **Save**.

E5061B Saving a Project from the E5061B Measurement Screen

1. Display the E5061B measurement screen following the instructions given in Switching to the E5061B Measurement Screen.
2. Open the Save As dialog box using the following key sequence:
   - **Macro Setup** > **Save Project**
3. The **Save As dialog box** appears. Specify the file name and location (drive or folder) and click Save.

Saving a Module (Exporting)

Alternatively, you can save each module (user form, standard, or class) of your VBA program individually. To save a module, you must use Visual Basic Editor. User forms are saved as .frm files, standard modules as .bas files, and class modules as .cls files.

a. In Project Explorer, click the file name that appears under the desired module icon to highlight it.

b. Open the Export File dialog box by doing one of the following:
    - On the **File** menu, click **Export File...**.
    - Click the right mouse button, and click **Export File...**.
    - Press **Ctrl** + **E** keys on the keyboard.
c. The **Export File dialog box** appears. Specify the file name and location (drive or folder) and click **Save**.

Loading a VBA Program

- Overview
- Loading a Project

Other topics about Operation Basics

Overview

Once you have saved a project or module file, you can load it later whenever necessary.

Loading a Project

You can load a saved project file either from the E5061B measurement screen or by specifying that the project file be automatically loaded when the power is turned ON.

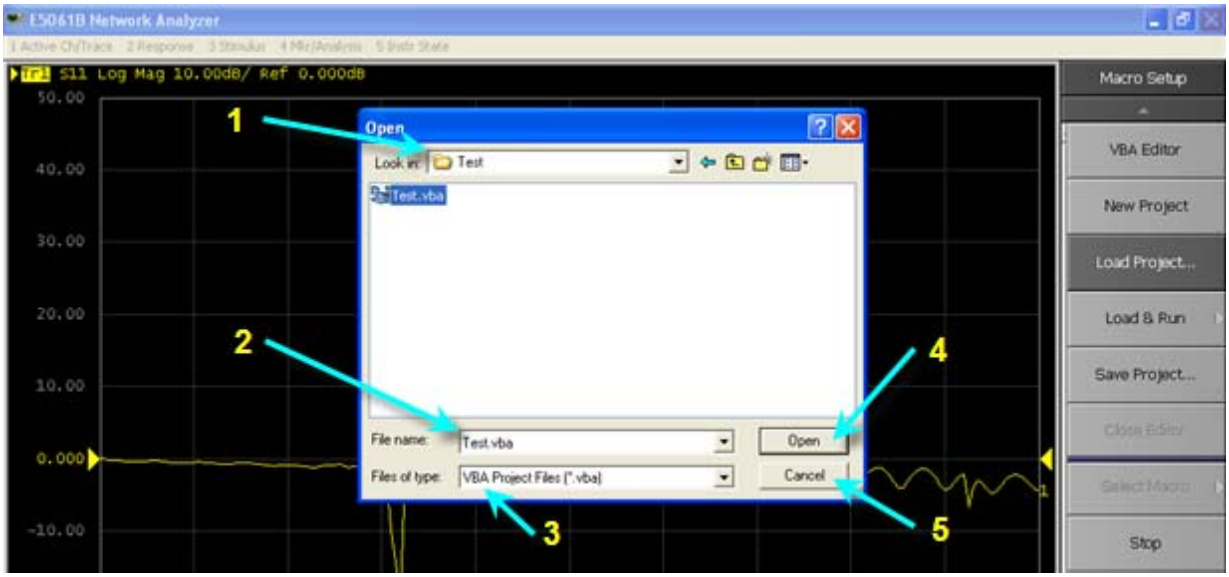
Loading a Project from the E5061B Measurement Screen

1. Press **Macro Setup** key, then click **Load Project**.

   NOTE When another project has already been loaded on the Visual Basic Editor, the message prompts to save the current project. If you want to save the project, click **Yes** button to display a dialog box for saving. For saving the project, see Saving a Project.

2. The Open dialog box appears. Specify the file name and location (drive or folder) of the file you want to load and click **Open**.

Open dialog box



The Open dialog box has the following user interface elements:

1. **Look in**: Specify the location (drive or folder) where the project resides.
2. **File name**: Specify the file name of the project you want to load.
3. **Files of type**: Select the type of the file you want to load. Normally, you should select **VBA Project Files [*.vba]**.
4. **Open**: Clicking this button loads the project.
5. **Cancel**: Clicking this button closes the Open dialog box and brings you back to the main screen.

Automatically Loading a Project at Power-On

Once you have saved a project file that satisfies the following conditions, the project loads automatically whenever the power is turned ON.

Auto-loaded project	Conditions
Directory where the project resides.	**A**:\ or **D**:\
Project file name	autoload.vba

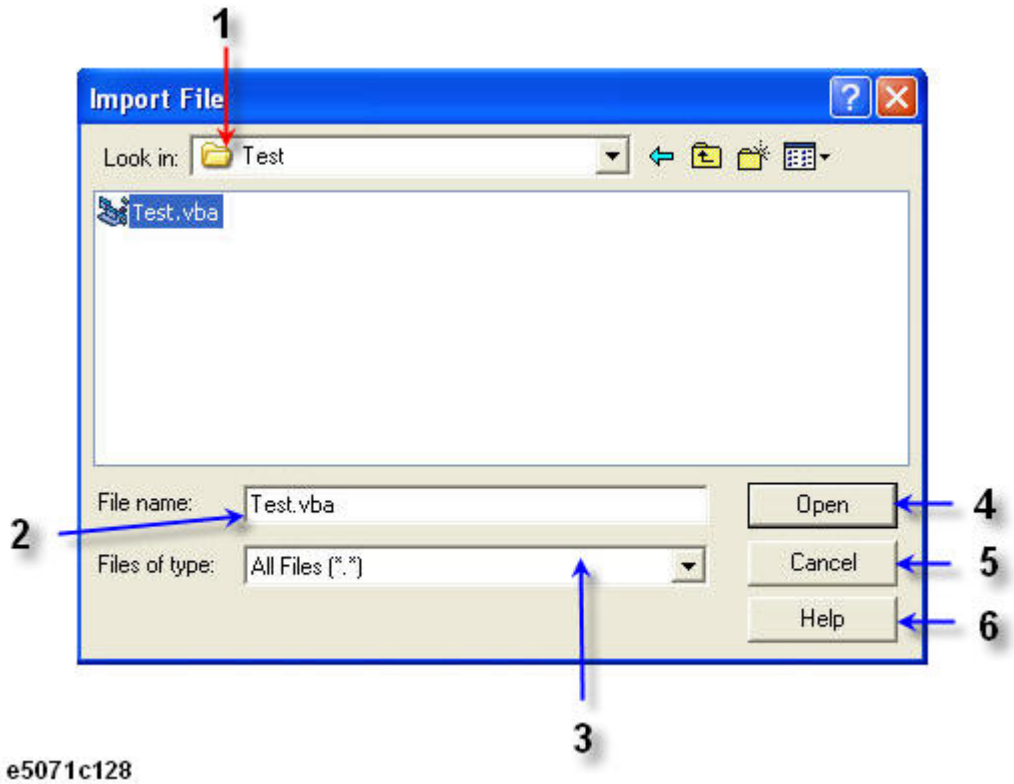
If there is the file named "autoload.vba" in both the A drive and the D drive, the file in the A drive is used.

Loading a Module (Importing)

To load a saved module into a project, you must use Visual Basic Editor.

1. In Project Explorer, click the file name that appears under the desired module icon to highlight it.
2. Open the Import File dialog box by doing one of the following:
   - On the **File** menu, click **Import File...**.
   - In Project Explorer, right-click the "VBAProject" icon, and click **Import File...**.
   - Press **Ctrl** + **M** keys on the keyboard.
3. The Import File dialog box appears. Specify the file name and location (drive or folder) of the file (module) you want to load and click **Open**.
4. The Import File dialog box has the following user interface elements:

Import File dialog box



e5071c128

The Import File dialog box has the following user interface elements:

1. **Look in**: Specify the location (drive or folder) where the module resides.
2. **File name**: Specify the file name of the module you want to load.
3. **Files of type**: Select the type of the file you want load. Normally, you should select **VB Files [*.frm,*.bas,*.cls]**.
4. **Open**: Clicking this button loads the module.
5. **Cancel**: Clicking this button closes the Import File dialog box and brings you back to the main screen.
6. **Help**: Clicking this button brings up VBA Online Help.

Running a VBA Program

- Overview
- Running a Previously Loaded VBA Program
- Running a Program from the_E5061B_Measurement_Screen
- Loading and Executing Programs in Batch Process

Other topics about Operation Basics

Overview

The E5061B provides 2 methods to execute a VBA program: executing a program that you loaded previously and loading and executing a program in a batch process. The execution status of the VBA program is indicated in the instrument status bar, as shown in the following figure. "Run" indicates that the program is running while "Stop" indicates that the program has stopped.

Instrument status bar indicating the status of the VBA program

Running a Previously Loaded VBA Program

Running a Program from Visual Basic Editor

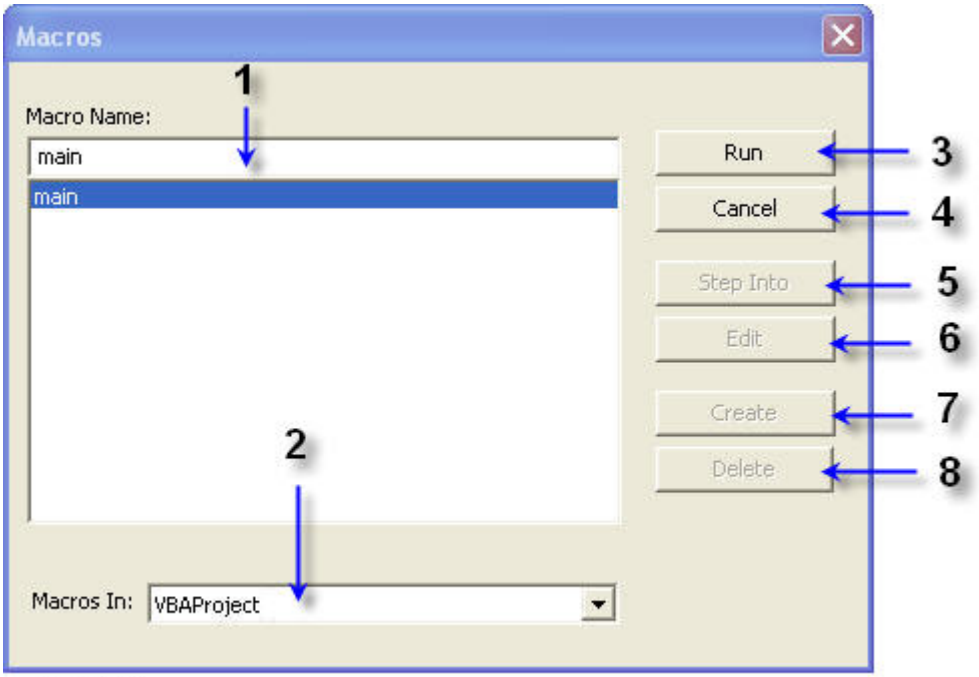
The E5061B allows you to run a previously loaded VBA program using one of the four methods listed below.

1. Open the Macros dialog box by doing either one of the following:
   - On the **Run** menu, click **Run Macro**.
   - On the **Tools** menu, click **Macros...**.
   - On the toolbar, click "Run Macro" icon.
   - Press **F5** key on the keyboard.

   NOTE Doing the above steps with the cursor positioned within a procedure in the code window runs the program immediately without displaying the Macros dialog box.

1. In the Macros dialog box, select the VBA program (procedure name) you want to run, and click the **Run** button.

Macros dialog box



e5071c133

The Macros dialog box has the following user interface elements:

1. **Macro Name**: Select the VBA program (procedure name) you want to run from the list box so that its name appears here.
2. **Macro In**: Specify the project that contains the VBA program you want to run. Normally, use the default.
3. **Run**: Clicking this button runs the selected VBA program (procedure).
4. **Cancel**: Clicking this button closes the Macros dialog box and brings you back to the main screen.
5. **Step Into**: Clicking this button brings up Visual Basic Editor and put it into step-in mode, where the selected VBA program is run step by step. This mode is primarily intended for use when debugging a VBA program. For more information on step-in mode, see Debug Toolbar.
6. **Edit**: Displays the code of the selected VBA program. You can use this for re-editing your code.
7. **Create**: This button is normally dimmed.
8. **Delete**: Clicking this button deletes the selected VBA program. Take care not to inadvertently delete your VBA program before saving it.

The Macros dialog provides access to subprograms (procedures enclosed between Sub and End Sub) created in a standard module.

Running a Program from the E5061B Measurement Screen

The E5061B allows you to run a program from E5061B screen using one of the four methods listed below.

1. Display the E5061B measurement screen following the instructions given in Switching to the E5061B Measurement Screen.
2. Run the VBA program (procedure) using the following key sequence:

- **Macro Setup** > **Select Macro - Module xxx**
  where "**Module**" is the object name (Name property shown in the property window) and "**xxx**" is the procedure name.
- Press the **Macro Run** key on the E5061B front panel. For a program to run from the measurement screen, its procedure name must be "Main" (subprogram enclosed between Sub Main() and End Sub), and its object name (Name property as displayed in the property window) must be "Module1".

  NOTE When you are working with the E5061B measurement screen, the E5061B's macro environment only provides access to those VBA programs that are created as subprograms (enclosed between Sub and End Sub) in a standard module.

Loading and Executing Programs in Batch Process

This section describes how to load and execute a program (VBA project) in a batch process by pressing the softkey corresponding to the program name.

1. Save the VBA program (VBA project file) into the following folder.
   D:\VBA

   NOTE This feature is available only for programs saved in **D:\VBA**. This feature is not available for programs saved in subfolders of **D:\VBA**.

   NOTE When copying a VBA program to D:\VBA from another folder, copy all the files necessary to execute the program to appropriate folders. When copying a factory-installed VBA program into D:\VBA, choose only its VBA project file.
2. Press **Macro Seup** key.
3. Click **Load & Run**.
4. Press the softkey corresponding to the VBA project file name of the program you want to execute. The pressed VBA project is loaded and the program of which the procedure name is set to "Main" (subprogram enclosed between Sub Main() and End Sub) and of which the object name (Name property as displayed in the property window) is set to "**Module1**" is executed.

NOTE There is no limit to the number of VBA project files that can be saved in **D:\VBA**. However, the maximum number of programs that can be displayed as softkeys is 50.

- File names of the VBA projects saved in **D:\VBA** are displayed as softkeys in alphabetical order.
- The maximum number of characters that can be displayed in a softkey is 12. If a file name has 13 or more characters, "..." is added to the 12th character from the beginning of the program name and displayed. In this case a .vba extension is omitted.

Stopping a VBA Program

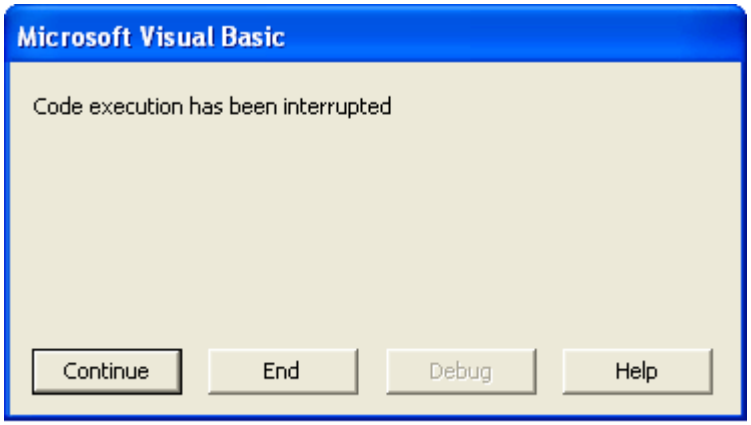
- Stopping a Procedure
- Abruptly Terminating a VBA Program

Other topics about Operation Basics

Stopping a Procedure

This section describes how to break a procedure during the execution of a VBA program.

1. To break the running VBA program, do one of the following:
   - On the **Run** menu, click **Break**.
   - On the toolbar, click "Break Macro" icon.
   - Press **Ctrl** + **Break** keys on the keyboard.
   - **Macro Setup** > **Stop** (E5061B measurement screen)
   - Press **Macro Break** key on the E5061B front panel.
2. A dialog box is displayed through forced interrupts, and the program is suspended.



e5071c250

Select one of the following:

- **Continue**: Resumes the execution of the program.
- **End**: Terminates the VBA program.
- **Debug**: Displays a run-time error.
- **Help**: Brings up VBA Online Help.

Abruptly Terminating a VBA Program

This section describes how to abruptly terminate a running procedure. When abruptly terminating the VBA program by the below methods, the "Program interrupted" message is shown on the instrument status bar at the bottom of the LCD display.

To terminate the running VBA program, perform one of the following:

- On the **Run** menu, click **Reset**.
- On the toolbar, click "Reset Macro" icon.
- Insert an *End* statement into your code.

Errors and Debugging

- Type of Errors
- Using a Debug Tool

Other topics about Operation Basics

Type of Errors

Errors in VBA programs are classified into the following two types:

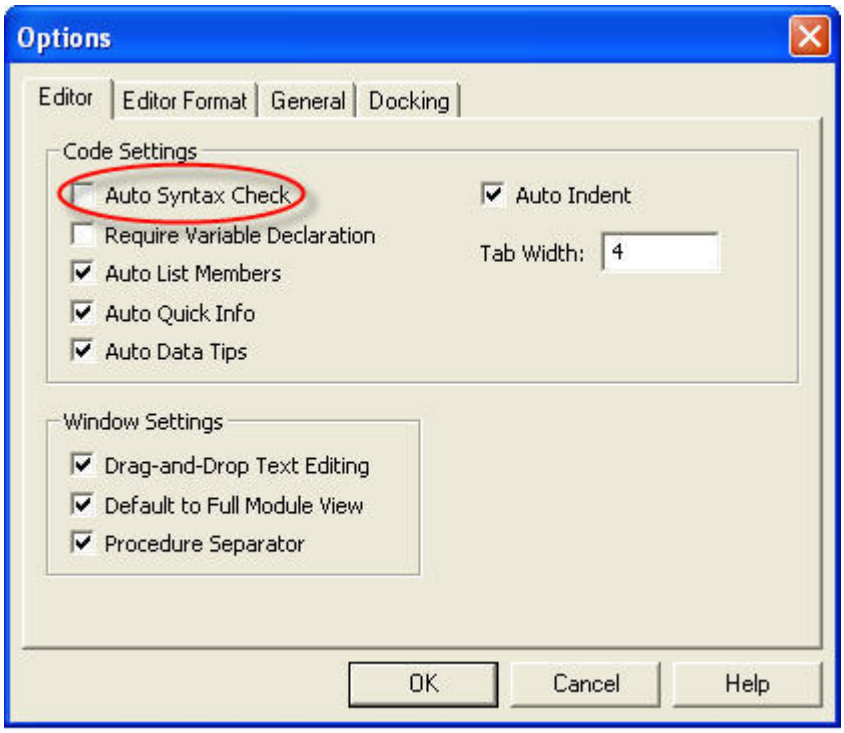
Syntax errors

A syntax error is generated when Visual Basic Editor detects an invalid statement that violates the Visual Basic syntax rules. For example, misspelled keywords generate syntax errors. An error dialog box appears that indicates the error message, and highlights the invalid statement in red. To get detailed information on the error, click the **HELP** button in the error dialog box to display the help topic on the error. You cannot run the macro until you correct the syntax error.

The E5061B VBA environment is configured by default to automatically check for syntax errors, but you can disable the auto syntax check feature using the following steps:

1. On the **Tools** menu, click **Options...**.

2. On the **Editor** tab, clear the **Auto Syntax Check** check box.



e5071c121

3. Click the **OK** button.

Run-time Errors

A run-time error is generated when a VBA program attempts to execute an invalid statement at run time. When a run-time error is generated, the program is stopped at the invalid statement, and an error dialog box appears. You can terminate the program by clicking the END button in the error dialog box. You can also click the DEBUG button in the error dialog box to identify the statement that caused the error. In this case, the statement in question is highlighted in yellow.

NOTE Some run-time errors occur under particular conditions, even though a program could run without getting errors under normal conditions. For example, the "Target value not found" error that occurs when a program that analyzes the results using the Marker Bandwidth Search feature, fails to perform bandwidth search because the marker is not in the appropriate position. The "Ecal module not in RF path" error that occurs when a program that performs calibrations using a ECal module, fails to measure the calibration data because the ECal module is not appropriately connected to test ports, and so on.

Using a Debug Tool

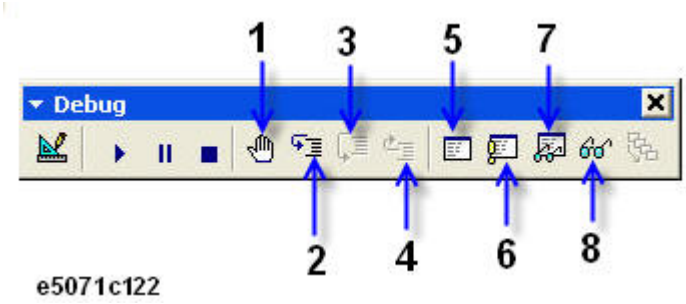
The E5061B's VBA environment provides a variety of debug tools that help you identify logical errors. Detailed information on using the debug tools is covered in VBA Online Help and books on VBA.

Debug Toolbar

The debug toolbar provides tool buttons that allow you to easily access various debug tools.

On the **View** menu, click **Toolbars** > **Debug**.

Debug toolbar



1. Set/clear break points (keyboard **F9**): Places a break point at the cursor position or clears an existing break point.
2. Step-in (keyboard: **F8**): Runs the VBA program step by step. If the current program contains a call to another procedure, that procedure is also run step by step.
3. Step-over (keyboard: **Shift** + **F8**): Runs the VBA program step by step. If the current program contains a call to another procedure, that procedure is run as one line.
4. Step-out (keyboard: **Ctrl** + **Shift** + **F8**): Executes the remaining lines of the function where the execution point is currently placed.
5. Local window: Opens the local window that shows the current values of local variables.
6. Immediate window (keyboard: **Ctrl** + **G**): Opens the immediate window that evaluates entered values of variables or expressions.
7. Watch window: Opens the watch window that displays the current value of a specified expression.
8. Quick window (keyboard: **Shift** + **F9**): Displays the current value of a specified expression in a dialog box.

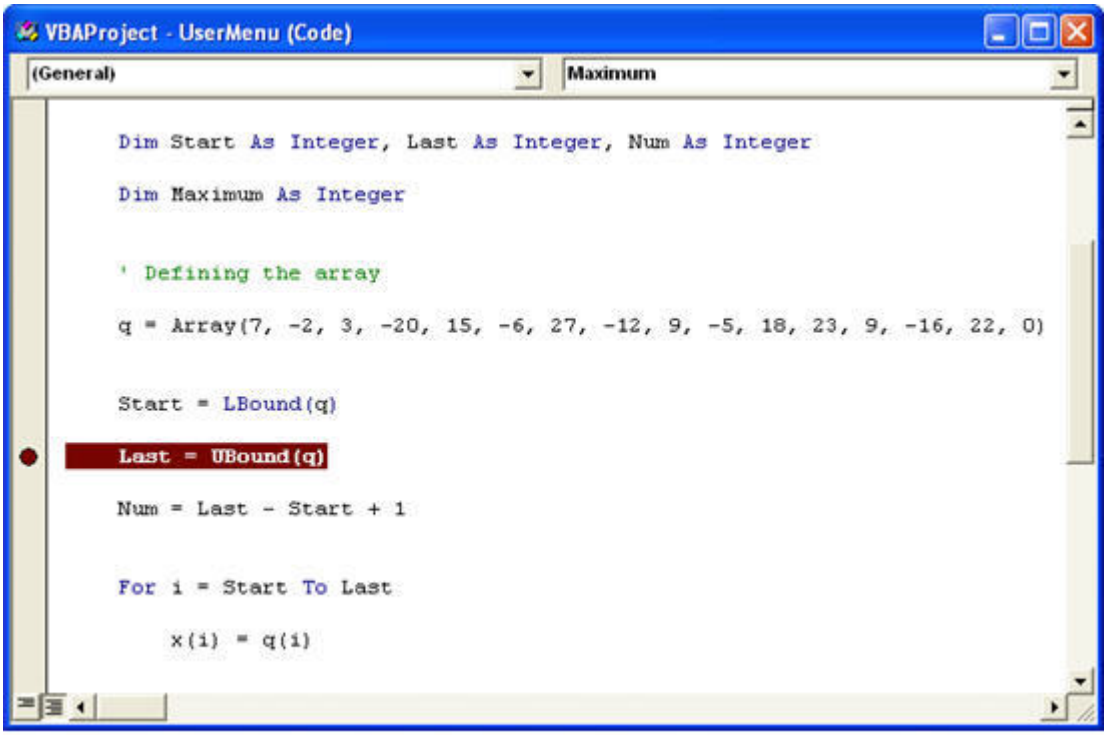
Setting a Break Point

By placing a break point at a particular statement in a VBA program, you can automatically suspend the program when it is executed to that statement.

When you put a break point at a line, the line is highlighted in umber as shown in the following figure. To set a break point, perform one of the following:

- Place the cursor at the desired line of code, and click the "Set/clear break points" button on the debug toolbar.
- Click anywhere in the margin indicator bar of the code window.

Setting a break point



e5071c123

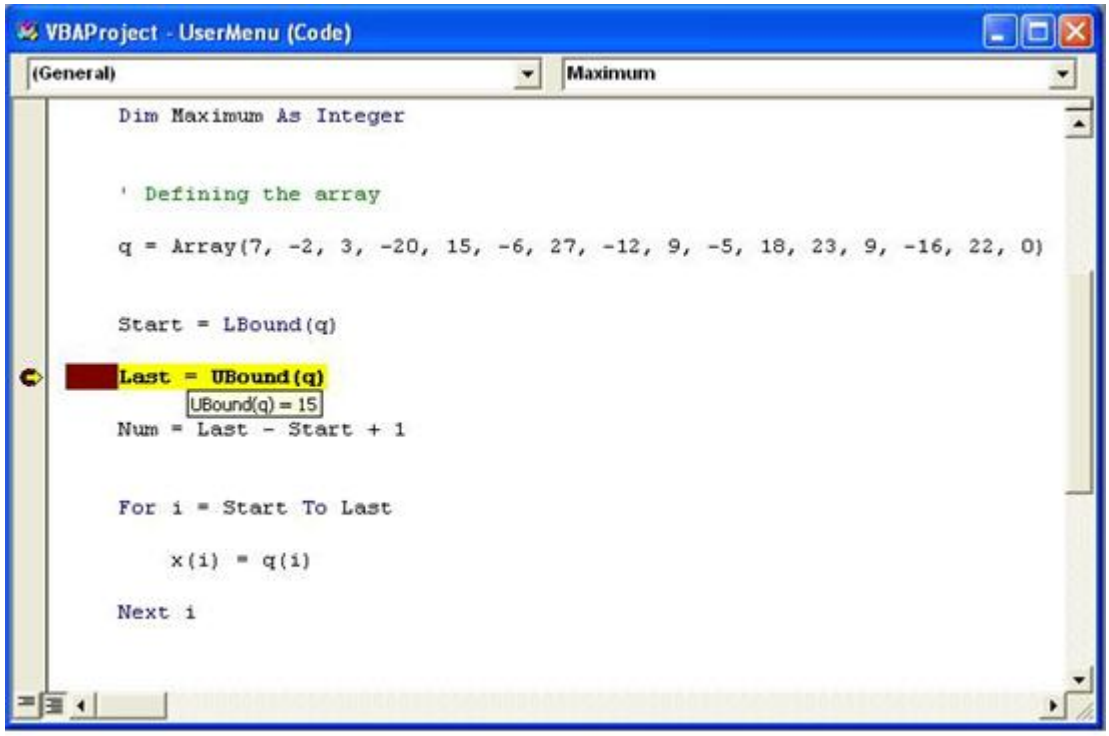
Monitoring Variable or Property Values

With your VBA program suspends, you can use the following debug tool to monitor variables or properties. To do this, you must set a break point, run the VBA program, and suspend it.

Data Hint

When you point to the variable or expression of interest, Data Hint shows the current value as shown in the following figure.

Data Hint



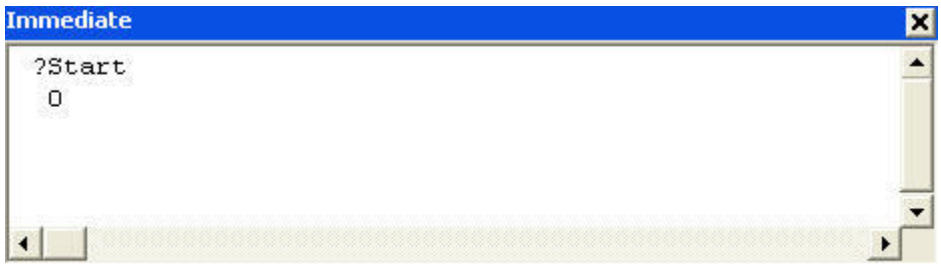
e5071c127

Immediate Window

To display the immediate window, click  on the debug toolbar.

In the immediate window, enter a question mark (?) followed by the variable or expression whose value you want to check, and press the Enter key, as shown in the following figure. The current value appears in the line that follows.

Immediate window

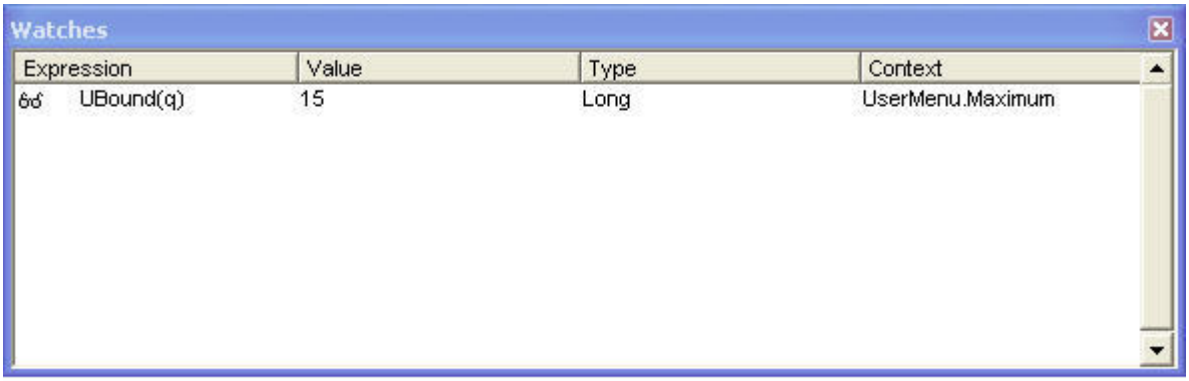


e5071c124

Watch Window

To display the watch window, click the "Watch Window" button on the debug toolbar.

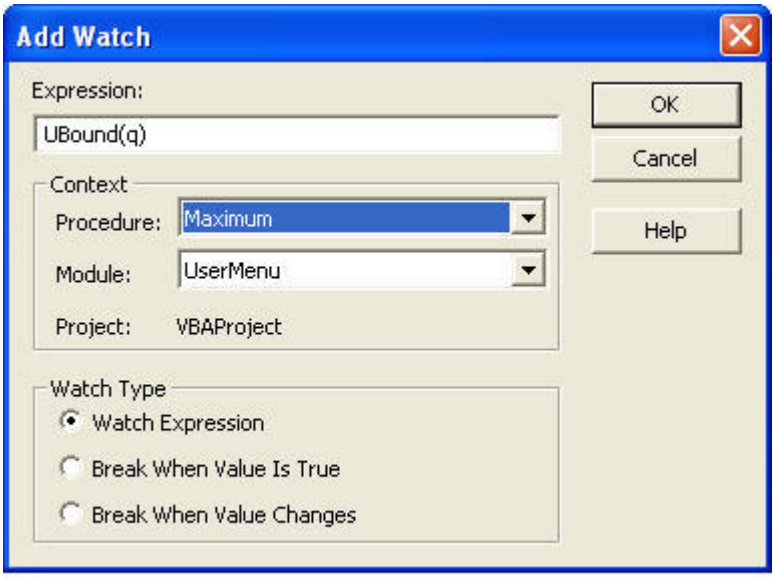
Watch window



e5071c126

1. On the **Debug** menu, click **Add Watch…**. to open the Add Watch dialog box.
2. As shown in the following figure, you can specify an expression of interest as a watch expression to always monitor its value.
3. Click the **OK** button.

Add Watch dialog box



e5071c125

Quick Watch

In the code window, select a variable or expression whose value you want to watch. On the debug toolbar, click the "Quick Watch" button to open the Quick Watch dialog box. The dialog box displays the current value of your specified variable or expression.

Also, you can click the **Add** button in the Quick Watch dialog box to specify the current expression as a watch expression.

Printing Output Values in the Echo Window

Other topics about Operation Basics

Overview

The echo window, which appears at the lower section of the E5061B measurement screen, can be used to display a message or the return value (data) of an object.

Entering Values Output to Echo Window

You can use the COM objects listed below to enter values output to the echo window.

- ECHO
- SCPI.DISPlay.ECHO.DATA

Opening Echo Window

You can use the COM objects listed below to open the echo window.

- SCPI.DISPlay.TABLe.TYPE
- SCPI.DISPlay.TABLe.STATe

Alternatively, you can also open the echo window using the following key sequence:

Macro Setup > **Echo Window (ON)**

Clearing Values Output from Echo Window

You can use the COM object shown below to clear values output from the echo window.

- SCPI.DISPlay.ECHO.CLEar

Alternatively, you can also clear values output from the echo window using the following key sequence:

Macro Setup > **Clear Echo**

Uses Advanced Techniques

- Accessing a List of E5061B COM Objects
- Using Automatic Library References

Other topics about Operation Basics

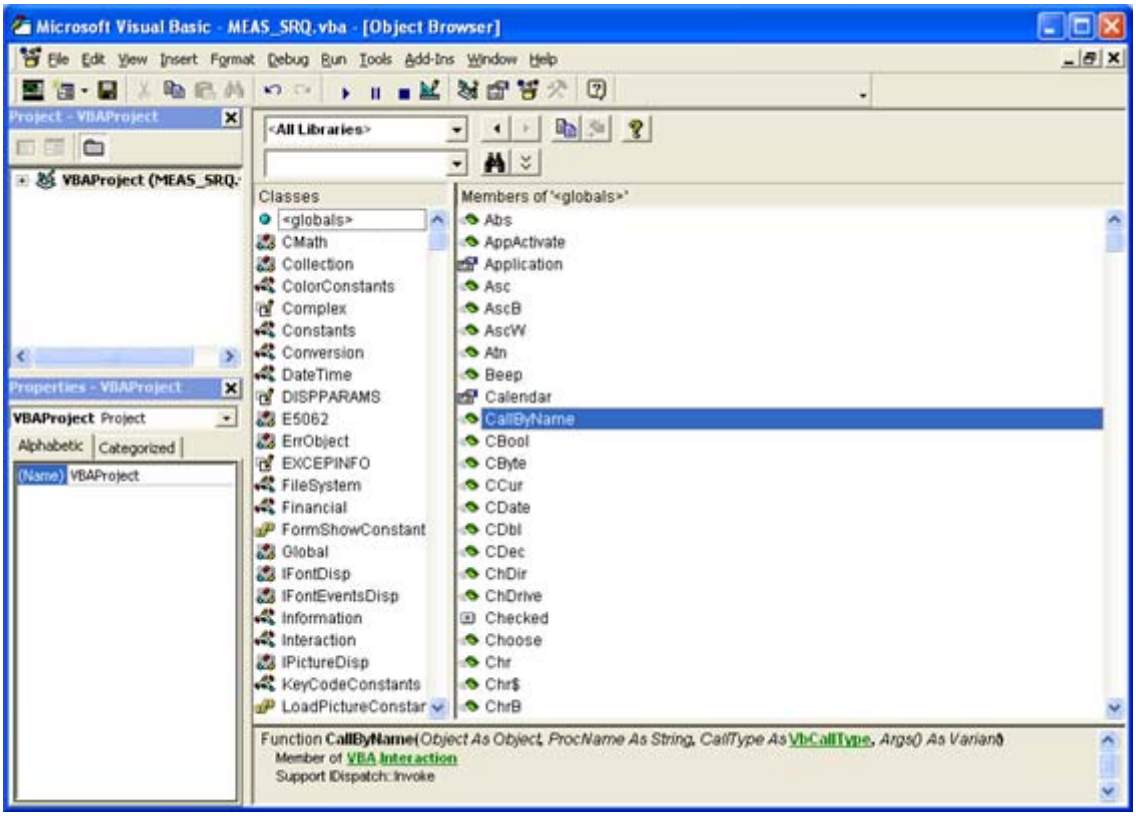
Accessing a List of E5061B COM Objects

The E5061B VBA environment provides COM objects that support the control of E5061B. When you are developing a program using E5061B COM objects, you can access a list of E5061B COM objects by opening Object Browser within Visual Basic Editor.

1. To open Object Browser, do one of the following:
   - On the **View** menu, click **Object Browser**.
   - On the toolbar, click "Object Browser" icon.
2. Select **E5062Lib** from the Project/Library box to display the E5061B library as shown in the following figure.

   NOTE There are some COM objects NOT used in controlling with E5061B VBA in the list of the E5061B COM objects displayed on the Object Browser. The COM objects NOT used in controlling with E5061B VBA are not described in the COM object reference.

How to use Object Browser



Using Automatic Library References

For libraries that satisfy the following conditions, the library reference automatically sets whenever a new project is created and loaded (**Macro Setup** > **New Project**).

Automatically referenced libraries	**Conditions**
Directory where the library resides.	**D:\Agilent**
Extensions of libraries	olb, tlb, dll, or ocx

To check the library reference setting, you must use Visual Basic Editor.

Follow these steps to check the library reference setting.

- On the **Tools** menu, click **References...**.

  NOTE A project sets the library reference when the project is created. Therefore, if the existing project is loaded, libraries

added after the development of the project are not automatically set in the library reference.

Using VBA Online Help

- Overview
- Accessing VBA Online Help

Other topics about Operation Basics

Overview

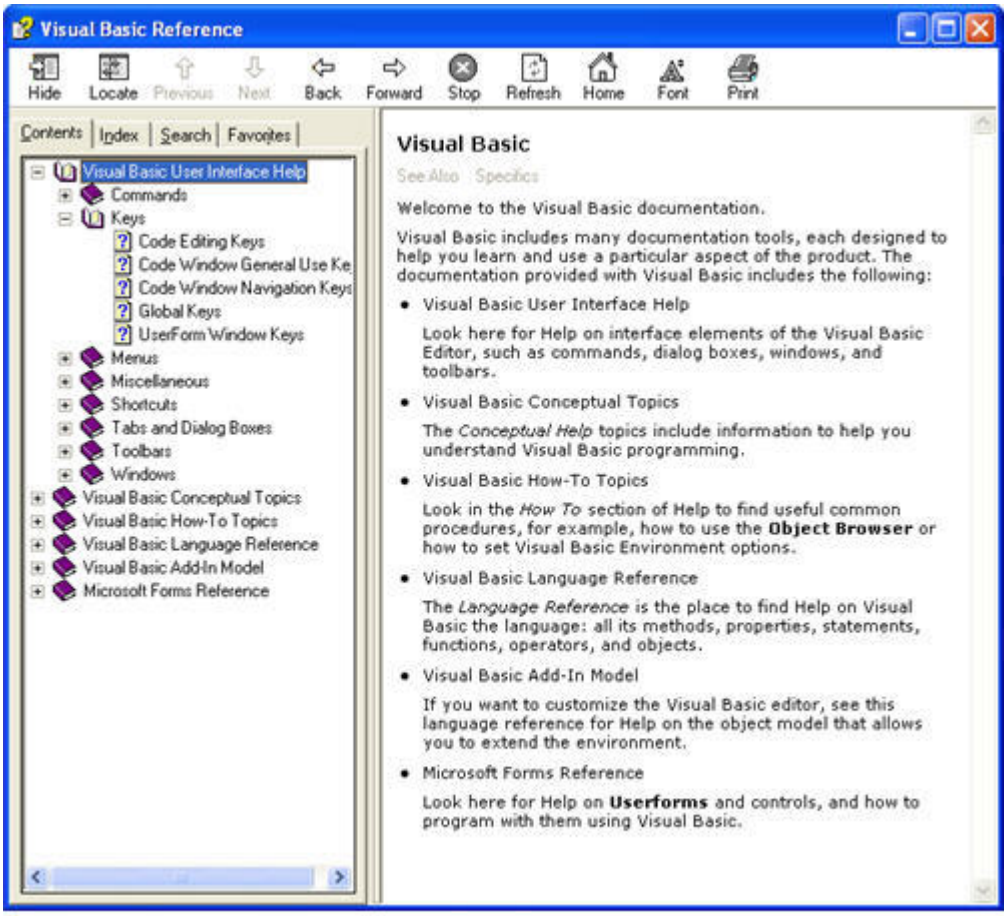
VBA Online Help provides useful topics, such as the VBA terminology or how to use a particular feature. In VBA Online Help, you can find a topic of interest through the Contents or by entering specific keywords.

Accessing VBA Online Help

From Visual Basic Editor, do one of the following to access the VBA Online Help screen.

- On the **Help** menu, click **Microsoft Visual Basic Help**.
- Press **F1** key on the keyboard.
- On the toolbar, click "VBA Help" icon.

VBA Online Help screen



e5071c135

Using the Contents Tab

Clicking the **Contents** tab in the VBA Online Help screen brings up the items listed below. The E5061B VBA Online Help has a hierarchical table of contents. Click an item to expand it, and then find a topic of interest.

- Visual Basic User Interface Help
- Visual Basic Conceptual Topics
- Visual Basic How-To Topics
- Visual Basic Language Reference
- Visual Basic Add-In Model
- Microsoft Forms Reference

When you need information on using Visual Basic Editor, use User Interface Help and How-To Topics as primary sources of information. Format of VBA program is covered in Visual Basic Conceptual Topics. Properties and methods supported by VBA are covered in Visual Basic Language Reference and Visual Basic Add-In Model. Information on using user forms is covered in Microsoft Forms Reference.

Using the Index Tab

In the VBA Online Help screen, click the **Index** tab, and enter a keyword(s) into the text box. For example, you may wish to search for "Sub" or "With" when you are writing your own code.

Looking up a Keyword in the Code within Visual Basic Editor

When you want to know the usage or meaning of a keyword in a sample program or some other code, you can quickly access the help topic on that keyword by moving the cursor to the keyword and pressing **F1** key.

Controlling E5061B

Controlling E5061B

- Detecting the End of Measurement
- Reading/Writing Measurement Data
- Executing a Procedure with a Softkey (User Menu Function)

Detecting End of Measurement

- Overview
- Using Status Register
- Using SCPI.TRIGger.SEQuence.SINGle Object

Other topics about Controlling E5061B

Overview

This chapter uses sample programs to demonstrate how to trigger the instrument to start a new measurement cycle and how to detect the end of a measurement cycle. The trigger system is responsible for such tasks as detecting the start of a measurement cycle (triggering) and enabling/disabling measurement on each channel.

You can detect the end of measurement by using either the status register or the SCPI.TRIGger.SEQuence.SINGle object.

Using Status Register

The status of the E5061B can be detected through the status register. If your program is based on SPCI commands, you can use SRQ (Service Request) interruptions to detect the end of measurement.

On the other hand, if your program is based on COM objects, SRQ interruptions are not available; instead, you can use the following object to suspend the program until SRQs are generated upon completion of measurement.

- WaitOnSRQ

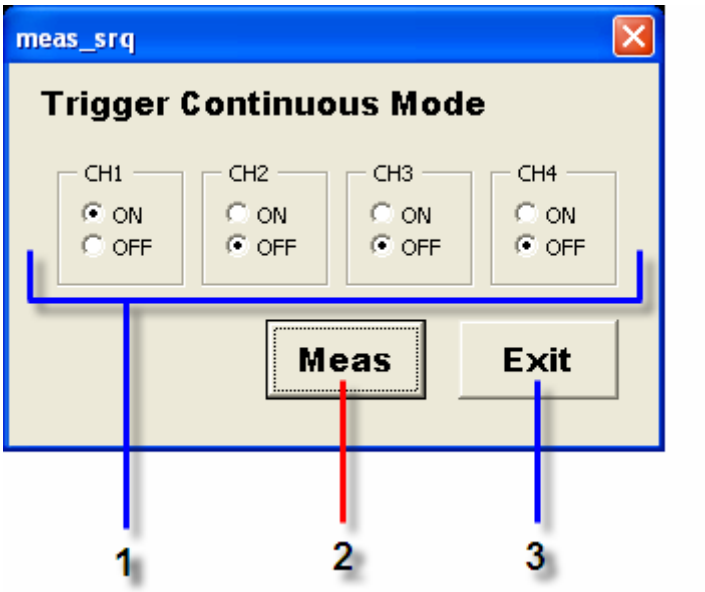
Sample program is available to download from the Agilent Support page, named **meas_srq.vba**. It demonstrates how to use the status register to suspend the program until the end of measurement. This VBA program consists of the following modules:

Object name	Module type	Content
frmSrqMeas	UserForm	Uses the status register to wait for the end of measurement.
mdlSrqMeas	Standard module	Invokes a UserForm.

NOTE This sample program correctly runs when the maximum number of channels/traces is set to 4 channels/4 traces.

When you run this VBA program, the following UserForm appears.

The UserForm when running the meas_srq.vba program



For how to use each element, see the following description.

1. The program turns ON/OFF Continuous Activation mode for each channel and determines whether to enable or disable each channel for measurement.
2. The program triggers the instrument to start a new measurement cycle, waits for the end of measurement, and then displays a message. For detail, see the description of the code window.
3. The program exits, and the UserForm disappears.

In Visual Basic Editor, open the UserForm (object name: frmSrqMeas), and double-click the **Meas** or **Exit** button to bring up the code window. The following is the description of the subprograms associated with the respective buttons.

Using SRQs to detect the end of measurement (object name: frmSrqMeas)

```
""
' Procedure called when the user clicks the Exit button on the UserForm.
""
private Sub cmdExit_Click()
'
' Unloads the UserForm from the memory, and terminates the program.
'
```

```
    Unload Me
'
End Sub
'
'
''''
' Procedure called when the user clicks the Meas button on the UserForm.
''''
Private Sub cmdMeas_Click()
    Dim Cond As Boolean
'
' Hides the UserForm (object name: frmSrqMeas) from the screen.
'
    frmSrqMeas.Hide
'
' Displays 4 channel windows.
'
    SCPI.DISPlay.Split = "d12_34"
'
' Sets the trigger source to "bus".
'
    SCPI.TRIGger.SEQuence.Source = "bus"
'
' These lines turn on or off Continuous Activation mode for each channel
' depending on whether the corresponding option buttons are on or off.
' By default, the mode is turned on for channel 1 only.
'
    SCPI.INITiate(1).CONTinuous = optOn1.Value
    SCPI.INITiate(2).CONTinuous = optOn2.Value
    SCPI.INITiate(3).CONTinuous = optOn3.Value
    SCPI.INITiate(4).CONTinuous = optOn4.Value
'
' These lines configure the instrument so that operation status event
' register's bit 4 is set to 1 only when operation status condition
' register's bit 4 is changed from 1 to 0 (negative transition).
'
    SCPI.STATus.OPERation.PTRansition = 0
```

```
    SCPI.STATus.OPERation.NTRansition = 16
'
' Enables the operation status event register's bit 4.
'
    SCPI.STATus.OPERation.ENABle = 16
'
' Enables the status byte register's bit 7.
'
    SCPI.IEEE4882.SRE = 128
'
' Clears the status byte register and operation status event register.
'
    SCPI.IEEE4882.CLS
'
' Triggers the instrument to start a measurement cycle.
'
    SCPI.IEEE4882.TRG
'
' Verifies that the instrument is in a measurement cycle, and suspends
' the program until the end of measurement.
' The time-out is set to 100 seconds (maximum value).
'
    WaitOnSRQ Cond, 100000
'
' These lines display a measurement completion message upon detecting
' the end of measurement.
'
    If Cond = True Then
      MsgBox "Measurement Completion"
    End If
'
' Displays the UserForm (object name :frmSrqMeas) on the screen.
'
    frmSrqMeas.Show
'
End Sub
```

Using the SCPI.TRIGger.SEQuence.SINGle Object

When you trigger the instrument by issuing the SCPI.TRIGger.SEQuence.SINGle object, you can use the SCPI.IEEE4882.OPC object to suspend the program until the end of measurement.

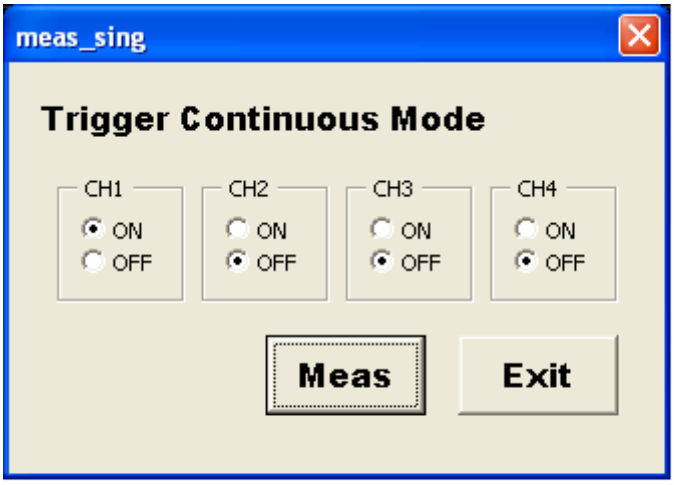
The sample program is available to download from the Agilent Support page, named **meas_sing.vba.** It demonstrates how to use the SCPI.TRIGger.SEQuence.SINGle object to suspend the program until the end of measurement. This VBA program consists of the following modules:

Object name	Module type	Content
frmSingMeas	UserForm	Uses the SCPI.TRIGger.SEQuence.SINGle and SCPI.IEEE4882.OPC objects to suspend the program until the end of measurement.
mdlSingMeas	Standard module	Invokes a UserForm.

NOTE This sample program correctly runs when the maximum number of channels/traces is set 4 channels/4 traces.

When you run this VBA program, a Userform appears.

The UserForm when running the meas_sing.vba program



In Visual Basic Editor, open the UserForm (object name:frmSingMeas), and double-click the **Meas** or **Exit** button to bring up the code window. The following is the description of the subprograms associated with the respective buttons.

Using the SCPI.TRIGger.SEQuence.SINGle object to suspend the program until the end of measurement (object name:frmSingMeas)

```
""
' Procedure called when the user clicks the Exit button on the UserForm.
""
Private Sub cmdExit_Click()
'
' Unloads the UserForm from the memory, and terminates the program.
'
    Unload Me
'
End Sub
'
'
""
' Procedure called when the user clicks the Meas button on the UserForm.
""
Private Sub cmdMeas_Click()
    Dim Dmy As Long
'
' Hides the UserForm (object name: frmSingMeas) from the screen.
'
    frmSingMeas.Hide
'
' Displays 4 channel windows.
'
    SCPI.DISPlay.Split = "d12_34"
'
' Sets the trigger source to "bus".
'
    SCPI.TRIGger.SEQuence.Source = "bus"
'
```

```
' These lines turn on or off Continuous Activation mode for each channel
' depending on whether the corresponding option buttons are on or off.
' By default, the mode is turned on for channel 1 only.
'
    SCPI.INITiate(1).CONTinuous = optOn1.Value
    SCPI.INITiate(2).CONTinuous = optOn2.Value
    SCPI.INITiate(3).CONTinuous = optOn3.Value
    SCPI.INITiate(4).CONTinuous = optOn4.Value
'
' Triggers the instrument to start a measurement cycle.
'
    SCPI.TRIGger.SEQuence.SINGle
'
' Executes the SCPI.IEEE4882.OPC object to suspend the program until
' the value of 1 is returned indicating the end of measurement.
'
    Dmy = SCPI.IEEE4882.OPC
'
' Displays a measurement completion message.
'
    MsgBox "Measurement Completion"
'
' Displays the UserForm (object name: frmSingMeas) on the screen.
'
    frmSingMeas.Show
'
End Sub
```

Reading / Writing Measurement Data

- Overview
- Sample Program

Other topics about Controlling E5061B

Overview

This section describes how to process the E5061B's internal data. You can use these internal data arrays: corrected data arrays, corrected memory arrays, formatted data arrays, formatted memory arrays, and stimulus data arrays. For more information on the internal data arrays, see Internal Data Processing.

To read/write a formatted data array, formatted memory array, corrected data array, or corrected memory array use the following objects:

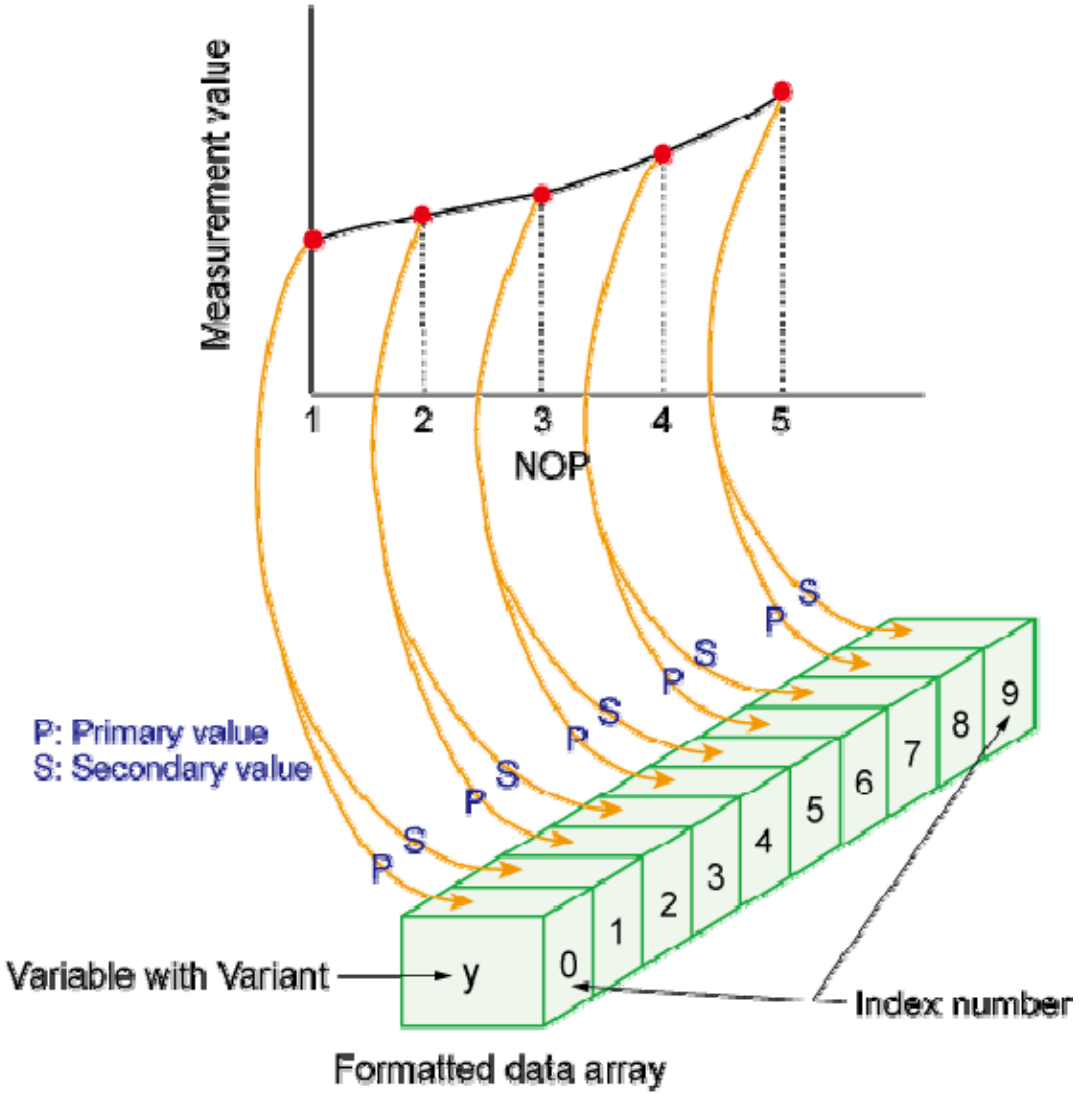
- SCPI.CALCulate(Ch).SELected.DATA.FDATa
- SCPI.CALCulate(Ch).SELected.DATA.FMEMory
- SCPI.CALCulate(Ch).SELected.DATA.SDATa
- SCPI.CALCulate(Ch).SELected.DATA.SMEMory

To read a stimulus data array, use the following objects:

- SCPI.SENSe(Ch).FREQuency.DATA

The E5061B VBA allows you to deal with multiple pieces of data through variables of Variant type. Variant variables can contain any type of data, allowing you to deal with array data without being aware of the number of elements. For example, a formatted data array that includes 5 measurement points is stored as shown in the following figure. Note that a formatted data array always contains 2 data items per measurement point, whichever data format is used. For more information on contained data, see Internal Data Processing. you can find a table that describes the relationship between contained data items and data formats.

Example storing data into a Variant variable



e5071c436

NOTE When you use one of the objects listed above, the base index number of the array is always 0 even if the declaration section contains the "Option Base 1" statement, which specifies the use of the base array index of 1.

For example, you may wish to read the formatted data array for a particular trace in its entirety (including all measurement points), display the data in the echo window, and then write the data into another trace. How to implement such a process can be better understood with the aid of a sample program.

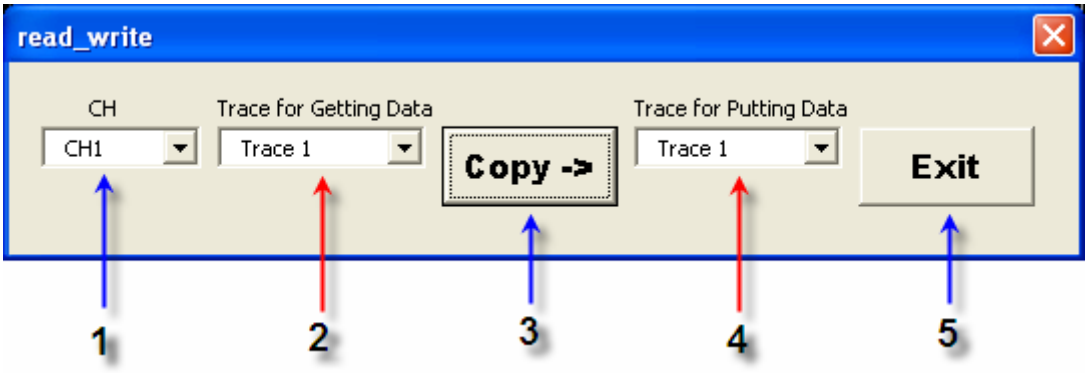
Sample program is available for download from the Agilent Support page, named "**read_write.vba**", that demonstrates how to read and write measurement data. This VBA program consists of the following modules:

Object name	Module type	Content
frmReadWrite	UserForm	Reads, displays, and writes a formatted data array.
mdlReadWrite	Standard module	Invokes a UserForm.

NOTE This sample program runs correctly when the maximum number of channels/traces is set to 4 channels/4 traces.

When you run this VBA program, a following window appears.

UserForm of read_write.vba program



The program lets the user specify the channel to be controlled.

1. The program lets the user specify which trace's formatted data array to read (source trace).
2. The program reads the formatted data array for the trace specified by the user, display the measurement results in the echo window, and write the data into the trace specified by the user. For detail, see the description of the code window.
3. The program lets the user specify which trace's formatted data array to overwrite (target trace).
4. The program exits, and the window disappears.

In Visual Basic Editor, open the UserForm (object name: frmReadWrite), and double-click the entire UserForm or the **Copy ->** or **Exit** button to bring up the code window. The following is the description of the subprograms associated with the respective buttons.

Sample Program

Reading/displaying/writing a formatted data array (read_write.frm)

```
'''
' Procedure called when the user clicks the Copy button on the UserForm.
'''
'
Private Sub cmdCopy_Click()
    Dim X As Integer, Y As Integer, Z As Integer, I As Integer
    Dim ActCh As Long, TrGet As Long, TrPut As Long
    Dim TrCont As Long, Nop As Long
    Dim FmtData As Variant, Freq As Variant
    Dim Fmt As String
'
' These lines identify the selected items in each list and store them into
' the variables TrGet, TrPut, and ActCh.
'
    X = cboCh.ListIndex
    ActCh = X + 1
    Y = cboGet.ListIndex
    TrGet = Y + 1
    Z = cboPut.ListIndex
    TrPut = Z + 1
'
' If the specified target trace is not displayed, these lines display that trace.
'
    TrCont = SCPI.CALCulate(ActCh).PARameter.Count
    If TrCont < TrPut Then
      SCPI.CALCulate(ActCh).PARameter.Count = TrPut
    End If
'
' These lines make active the specified trace (TrGet: source trace) in
```

```
' the specified channel(ActCh) and hold the sweep.
'
    SCPI.CALCulate(ActCh).PARameter(TrGet).SELect
    SCPI.INITiate(ActCh).CONTinuous = False
    SCPI.ABORt
'
' Reads the number of measurement points for the specified channel (ActCh)
' and stores that number into the Nop variable.
'
    Nop = SCPI.SENSe(ActCh).SWEep.POINts
'
' Reads the formatted data array for the active trace (source trace)
' and store the data into the FmtData variable.
'
    FmtData = SCPI.CALCulate(ActCh).SELected.Data.FDATa
'
' Reads the stimulus array for the specified channel (ActCh)
' and stores the data into the Freq variable.
'
    Freq = SCPI.SENSe(ActCh).FREQuency.Data
'
' Reads the data format for the active trace (source trace) and
' store it into the Fmt variable.
'
    Fmt = SCPI.CALCulate(ActCh).SELected.Format
'
' These lines display the echo window in the lower part of
' the LCD screen.
'
    SCPI.DISPlay.TABLe.TYPE = "ECHO"
    SCPI.DISPlay.TABLe.STATe = True
'
' The lines display, in the echo window, each point along with
' one measured value (the odd part of the index is always 0) and
' a frequency if the Fmt is "MLOG", "PHAS", "GDEL", "MLIN", "SWR", "REAL",
' "IMAG", or "UPH"; or along with two measured values and a frequency
' if Fmt returns any other string.
```

```
'
    Select Case Fmt
    Case "MLOG", "PHAS", "GDEL", "MLIN", "SWR", "REAL", "IMAG", "UPH"
      ECHO "Nop", "Frequency(GHz)", "Data"
      For I = 0 To Nop - 1
        ECHO I + 1, Freq(I) / 1000000000#, FmtData(2 * I)
      Next I
    Case Else
      ECHO "Nop", "Frequency(GHz)", "Data1", "Data2"
      For I = 0 To Nop - 1
        ECHO I + 1, Freq(I) / 1000000000#, FmtData(2 * I), FmtData(2 * I + 1)
      Next I
    End Select
'
' Makes active the specified trace (TrPut: target trace) in the specified
' channel(ActCh).
'
    SCPI.CALCulate(ActCh).PARameter(TrPut).SELect
'
' Writes the formatted data array (FmtData) into the active trace (target trace).
'
    SCPI.CALCulate(ActCh).SELected.Data.FDATa = FmtData
'
End Sub
'
'
""
' Procedure called when the user clicks the Exit button on the UserForm.
""
Private Sub cmdExit_Click()
'
' Unloads the UserForm from the memory, and terminates the program.
'
    Unload Me
'
End Sub
'
```

```
'
''''
' Procedure that initializes the UserForm
''''
Private Sub UserForm_Initialize()
'
' When the program is launched, these lines add each list item and set the default value
' for each list.
'
    With cboCh
      .AddItem "CH1"
      .AddItem "CH2"
      .AddItem "CH3"
      .AddItem "CH4"
    End With
'
    With cboGet
      .AddItem "Trace 1"
      .AddItem "Trace 2"
      .AddItem "Trace 3"
      .AddItem "Trace 4"
    End With
'
    With cboPut
      .AddItem "Trace 1"
      .AddItem "Trace 2"
      .AddItem "Trace 3"
      .AddItem "Trace 4"
    End With
'
    cboCh.ListIndex = 0
    cboGet.ListIndex = 0
    cboPut.ListIndex = 0
'
End Sub
```

Executing a Procedure with a Softkey (User Menu Function)

Other topics about Controlling E5061B

Overview

The E5061B lets you perform procedures assigned to specific softkeys (**Macro Setup** > **User Menu** > **Button 1/2/3/4/5/6/7/8/9/10**), without using user forms, when that softkey is pressed. This function is called the user menu function.

NOTE You do not have to execute any VBA program when using the user menu function.

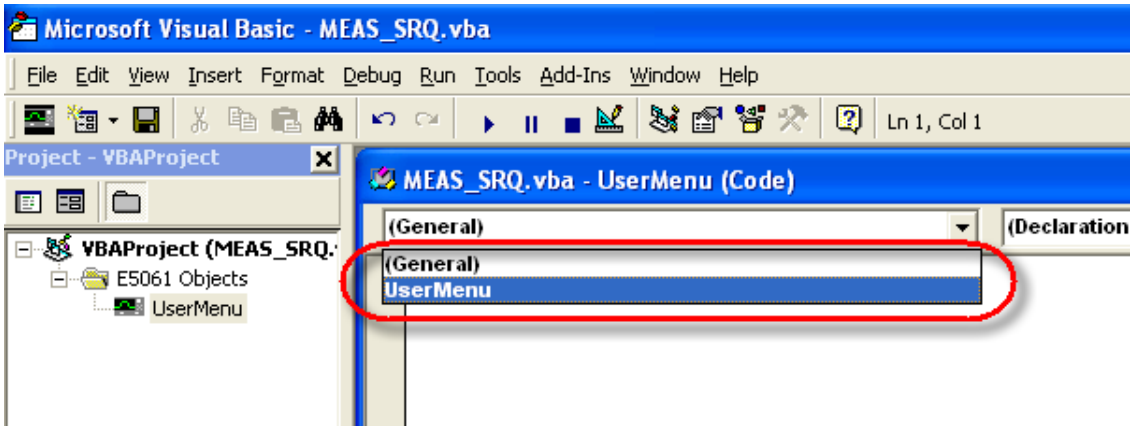
Preparing for User Menu Function

Before using the user menu function, perform the following preparation.

Coding of a Procedure Assigned to a Softkey

Follow these steps to create a procedure assigned to a specific softkey in the "UserMenu" object in the "E5061 Objects" folder.

1. Double-click UserMenu to open the code window.
2. In the object box in the code window, select **UserMenu** as shown below:



3. In the UserMenu_OnPress(ByVal Key_id As Long) procedure, create a program you want to assign to a specific softkey (specify with the *id* variable). For actual use example, see Line 70 to 430 in the Sample program using user menu UserMenu object.

NOTE During processing an event (during execution of a procedure for a key pressed), another event (an interrupt by a procedure for another softkey pressed) cannot be accepted.

NOTE You cannot save (export) the "UserMenu" object by module basis; save it by project basis.

Settings for Softkey Label and Softkey Enabled/Disabled

When you want to change the softkey labels for the user menu function, use the following COM object.

- UserMenu.Item(Key_id).Caption

When you want to set the softkey enabled/disabled for the user menu function, use the following COM object.

- UserMenu.Item(Key_id).Enabled

Moreover, when you want to preset the above settings for the user menu function, use the following COM object.

- UserMenu.PRESet

  NOTE The above user menu setting is also preset by pressing **Macro Setup** > **Preset User Menu** on the E5061B front panel.

Using User Menu Function

To execute a procedure assigned to a softkey, you need to generate an event of pressing the softkey. To generate an event, the manual method and the COM object method are available.

Method by Manual Operation

Click the specific softkey as follows:

Macro Setup > **User Menu** > **Button <No>.**

"**No.**" represents a button number. You can set the label for "**Button No.**" as you like. For detail, refer to the "Settings for Softkey Label and Softkey Enabled/Disabled." section.

Method by COM Object

You can use the following COM object to perform the same operation as pressing a specific softkey.

- UserMenu.Press(Key_id)

Sample Program

Sample program is available to download from the Agilent Support page, named **meas_user.vba**, that demonstrates how to use the user menu

function. This VBA program consists of the following standard module and the "UserMenu" object.

Object name	Module type	Content
mdlUserMenu	Standard module	Sets the softkey labels and enables interrupts from the softkeys.

The program (object name: mdlUserMenu) is described in detail in below code:

Sample program using user menu (object name: mdlUserMenu)

```
""
' This VBA program consists of the standard module and the "UserMenu" object.
""
'
' A common variable is declared.
'
   Public State As Boolean
'
""
' The program (object name: mdlUserMenu) is set the softkey labels
' and enables interrupts from the softkeys.
""
'
Sub Main()
   Dim I As Long, J As Long
'
' Stores True into the State variable.
'
   State = True
'
' Sets the first to third softkey (id: 1 to 3) enabled, and sets
' the fourth to tenth softkey (id: 4 to 10) disabled.
'
   For I = 1 To 3
     UserMenu.Item(I).Enabled = True
   Next I
```

```
'
    For J = 4 To 10
      UserMenu.Item(J).Enabled = False
    Next J
'
' Sets the first softkey label (id: 1) to "Setup", the second softkey
' label (id: 2) to "Meas", the third softkey label (id: 3) to "Exit".
'
    UserMenu.Item(1).Caption = "Setup"
    UserMenu.Item(2).Caption = "Meas"
    UserMenu.Item(3).Caption = "Exit"
'
' Displays the buttons for the user menu function in the softkey area.
'
    UserMenu.Show
'
' Processing repeated until the State variable is True (State = True).
' Detects an event that a specific softkey is pressed and
' enables the interrupt from the event.
'
    Do While State
      DoEvents
    Loop
'
End Sub
```

UserMenu object

The procedures of the "UserMenu" object are described in detail in below code.

Sample program using user menu ("UserMenu" object)

```
""
' Procedure called when the specific softkey is pressed.
""
'
Private Sub UserMenu_OnPress(ByVal id As Long)
    Dim I As Integer
    Dim Nop As Long, Dmy As Long
```

```
    Dim FmtData As Variant
'
    If id = 1 Then
'
' The procedure when the first softkey (id: 1) is pressed.
'
' Returns the E5061B to the preset state.
'
      SCPI.SYSTem.PRESet
'
' For channel 1, sets the sweep start value to 1.73 GHz,
' the sweep stop value to 1.83 GHz, and the number of
' measurement points to 51.
'
      SCPI.SENSe(1).FREQuency.STARt = 1730000000#
      SCPI.SENSe(1).FREQuency.STOP = 1830000000#
      SCPI.SENSe(1).SWEep.POINts = 51
'
' After aborting the measurement, sets the trigger source
' to the bus trigger and turns on the continuous trigger startup mode
' for channel 1.
'
      SCPI.ABORt
      SCPI.TRIGger.SEQuence.Source = "BUS"
      SCPI.INITiate(1).CONTinuous = True
'
' Displays the buttons for the user menu function in the softkey area.
'
      UserMenu.Show
'
    ElseIf id = 2 Then
'
' The procedure when the second softkey (id: 2) is pressed.
'
' Generates a trigger to start a single sweep and waits until the measurement
' finishes (1 is read out with the SCPI.IEEE4882.OPC object).
'
```

```
    SCPI.TRIGger.SEQuence.SINGle
    Dmy = SCPI.IEEE4882.OPC
'
' Retrieves the number of points in channel 1 and stores that number
' into the Nop variable.
'
    Nop = SCPI.SENSe(1).SWEep.POINts
'
' Specifies trace 1 of channel 1 to the active trace, retrieves the formatted
' data array, and stores the data into the FmtData variable.
'
    SCPI.CALCulate(1).PARameter(1).SELect
    FmtData = SCPI.CALCulate(1).SELected.DATA.FDATa
'
' Displays the echo window in the lower part of the LCD screen.
'
    SCPI.DISPlay.TABLe.TYPE = "ECHO"
    SCPI.DISPlay.TABLe.State = True
'
' Displays 2 measurement data values (primary value and secondary value)
' for each measurement point in the echo window.
'
    For I = 1 To Nop - 1
      ECHO FmtData(2 * I - 2), FmtData(2 * I - 1)
    Next I
'
  ElseIf id = 3 Then
'
' The procedure when the third softkey (id: 3) is pressed.
'
' Displays a program closing message, and stores False into the state
' variable to terminate the main program.
'
    MsgBox "Program ended"'
    State = False
'
  End If
```

```
'
End Sub
```

User Defined Variable

The E5061B is having an area in which a User can set any value. These area are divided by the different data format of the values. A maximum of ten (1 to 10) such areas can be used by each command.

For example, after setting the value (data) obtained using VBA of the E5061B to the User defined variable, this value is available to an external controller (program) through the use of these User defined variables.

NOTE Turning E5061B power ON/OFF initializes the User defined variables. They are not initialized through executing Preset.

These commands does not refers to or change the results of the E5061B.

- SCPI.PROGram.VARiable.ARRay(Vnum).DATA
- SCPI.PROGram.VARiable.ARRay(Vnum).SIZE
- SCPI.PROGram.VARiable.DOUBle(Vnum).DATA
- SCPI.PROGram.VARiable.LONG(Vnum).DATA
- SCPI.PROGram.VARiable.STRing(Vnum).DATA

Controlling Peripherals

Controlling Peripherals

- Overview
- Programming with VISA

Overview

Other topics about Controlling Peripherals

Overview

The E5061B macro function (E5061B VBA) can be used not only to automate measurements but also to control external measurement instruments connected via USB/GPIB interface by acting as a self-contained system controller (see An Overview of a Control System Based on the Macro Function).

The E5061B macro function (E5061B VBA) performs communications via the COM interface when controlling the E5061B itself, and it communicates via VISA (Virtual Instrument Software Architecture) when controlling external measurement instruments.

Preparation

Importing Definition Files

To use the VISA library in the E5061B macro (E5061B VBA), you need to import two definition files into your project with the Visual Basic editor to define the VISA functions and perform other tasks. The definition files are stored on the sample programs disk under the following filenames (for information on importing modules, refer to Saving a Module (Exporting).

- **visa32.bas**
- **vpptype.bas**

Programming with VISA

Other topics about Controlling Peripherals

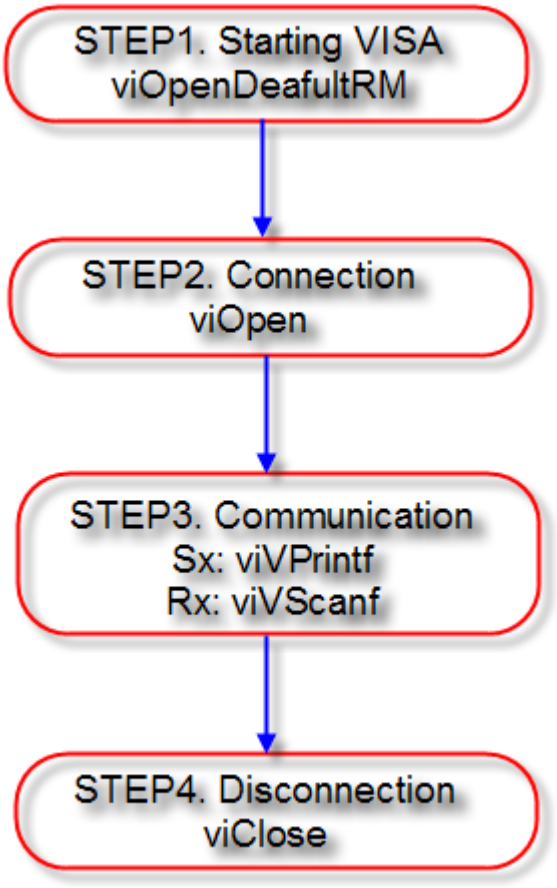
Overview

The following figure shows the flow of controlling the instrument with VISA. When developing a VISA program in the Visual Basic language, a special programing notice (in the readme text file listed below) must be reviewed.

For details on the use of the VISA library and the programing notice for using the VISA library with the E5061B macro (E5061B VBA), refer to the following files contained in I/O library CD-ROM.

- **visa.hlp** (on-line help for the VISA library)
- **vbreadme.txt** (notes on using the VISA library with VB)

Flow of instrument control with VISA



e5071c183

STEP 1. Starting VISA

The VISA system startup session is viOpenDefaultRM function in the sample program **ctrl_ext.vba**. VISA's viOpenDefaultRM function initializes and starts up the VISA system. The viOpenDefaultRM function must be executed before other VISA functions are called, and the parameter of this function has the startup information (Defrm in **ctrl_ext.vba**).

Syntax

viOpenDefaultRM(*param*)

Parameter

Parameter	(*param*)
Description	Startup information (output)
Data type	Long integer type

STEP 2. Connection

The connection session is viOpen function. VISA's viOpen function makes connection with the specified instrument. The viOpen function returns a

value so that the VISA functions can apply it to the specified instrument. The parameters of this function contain the startup information (Defrm), the address information of the specified instrument ("GPIB0::17::INSTR" in **ctrl_ext.vba**), access mode (0 in **ctrl_ext.vba**), timeout (0 in **ctrl_ext.vba**), and connection information (Equip in **ctrl_ext.vba**).

Syntax

viOpen(*param1*, *param2*, *param3*, *param4*, *param5*)

Parameters

Parameter	(*param1*)
Description	Startup information (input)
Data type	Long integer type

Parameter	(*param2*)
Description	Address information of the specified instrument (input)
Data type	Character string type
Syntax	"GPIB0::*gpib address*::INSTR" "USB0::*manufacturer ID*::*model code*::*serial number*::0::INSTR" (ex. "USB0::2391::2312::MY12345678::0::INSTR") "TCPIP0::*IP address*::inst0::INSTR"

Parameter	(*param3*)
Description	Access mode (Enter 0)

Parameter	(*param4*)
Description	Timeout (Enter 0)

Parameter	(*param5*)
Description	Connection information (output)

Data type	Long integer type

STEP 3. Communication

The communication session is viVPrintf function. VISA's viVPrintf function sends a program message (GPIB command) to the specified instrument. The parameters of this function contain the connection information (Equip), the program message (*IDN?), and the variable to be formatted (0 in **ctrl_ext.vba**).

NOTE To input/output GPIB commands, the viVPrintf function and the viVScanf function are mainly used, but other VISA functions are also available. For more information, refer to **visa.hlp** (online help for the VISA library).

Syntax

viVPrintf(*param1*, *param2*, *param3*)

Parameters

Parameter	(*param1*)
Description	Connection information (input)
Data type	Long integer type

Parameter	(*param2*)
Description	Program message (input) When sending a program message of the GPIB command, a message terminator is required at the end of the message (Chr$(10) in **ctrl_ext.vba**)
Data type	Character string type

Parameter	(*param3*)
Description	A variable to be formatted. If not applicable, enter 0.
Data type	Specified data type

The receiving session is viVScanf function. VISA's viVScanf function receives the result from the specified instrument and stores it in the output variable. The parameters of this function contain the connection information (Equip in **ctrl_ext.vba**), the format parameter for the output variable (%t in **ctrl_ext.vba**), and the output variable (Prod in **ctrl_ext.vba**).

Syntax

viVScanf(*param1*, *param2*, *param3*)

Parameters

Parameter	(*param1*)
Description	Connection information (input)
Data type	Long integer type

Parameter	(*param2*)
Description	Format parameter for the output variable
Data type	Character string type

Parameter	(*param3*)
Description	Output variable (output)
Data type	Character string type

STEP 4. Disconnection

The disconnection session is viClose function. VISA's viClose function disconnects communication and terminates the VISA system. The parameter of this function has startup information (Defrm in **ctrl_ext.vba**).

Syntax

viClose(*param*)

Parameter

Parameter	(*param*)
Description	Startup information (input)
Data type	Long integer type

Sample Program to Read Out the Product Information of Peripheral (Instrument)

The **ctrl_ext.vba** is a sample program that controls instruments connected through USB/GPIB interface cable using the E5061B as the system controller. This VBA program consists of the following modules.

Object name	Module type	Content
mdlCtrlExt	Standard module	Reads out the product information of external instrument.
Visa32	Standard module	Definition file to use VISA library (Visa32.bas)

NOTE When you control peripherals from E5061B VBA, use the GPIB commands provided for the instrument to communicate with VISA. On the other hand, when you control the E5061B itself with E5061B VBA, use the COM objects provided for the E5061B to communicate.

Read out the product information (ctrl_ext.vba)

```
""
' This program is sample of controlling peripherals.
""
'
Sub Main()
   Dim status As Long
   Dim Defrm As Long
   Dim Equip As Long
   Dim Prod As String * 100
'
' Initializes and starts up the VISA system and outputs the startup information to the Defrm variable.
' During this process, if an error occurs, the program goes to the error handling routine.
'
   status = viOpenDefaultRM(Defrm)
   If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
'
' Establishes the connection to the external instrument (GPIB address: 17) connected via GPIB and
' outputs the connection information to the Equip variable. During this process, if an error occurs,
' the program goes to the error handling routine.
'
   status = viOpen(Defrm, "GPIB0::17::INSTR", 0, 0, Equip)
   If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
'
```

```
' Queries the product information of the external instrument connected via USB/GPIB interface cable
' using VISA. During this process, if an error occurs, the program goes to the error handling routine.
'
    status = viVPrintf(Equip, "*IDN?" & Chr$(10), 0)
    If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
'
' Retrieves the product information through VISA and outputs it into the Prod variable. Displays the
' read-out result in the message box. During this process, if an error occurs, the program goes to
' the error handling routine.
'
    status = viVScanf(Equip, "%t", Prod)
    If (status <> VI_SUCCESS) Then GoTo VisaErrorHandler
    MsgBox Prod
'
' Breaks the communication and terminates the VISA system.
'
    Call viClose(Defrm)
'
    GoTo Prog_end
'
' If an error occurs in a VISA function, displays the detail of the error and terminates the program.
'
VisaErrorHandler:
    Dim VisaErr As String * 200
    Call viStatusDesc(Defrm, status, VisaErr)
    MsgBox "Error : " & VisaErr, vbExclamation
    Exit Sub
'
Prog_end:
'
End Sub
```

Application Programs

Application Programs

- Basic Measurement (measuring a band-pass filter)
- Connecting Hard Disk of External PC (shared folder)

Basic Measurement (measuring a band-pass filter)

- Overview
- Overview of the Program
- Description of the Program

Other topics about Application Programs

Overview

The **apl_bsc.vba** shows a sample program (VBA program) that demonstrates how to perform the basic measurement of the band-pass filter. This VBA program consists of the following standard module.

Object name	Module type	Content
mdlBscMeas	Standard module	Performs basic measurement of band-pass filter

Overview of the Program

The sample program performs full 2-port calibration using the 85032F calibration kit, measures a band-pass filter (center frequency: 947.5 MHz), and calculates and displays its bandwidth, insertion loss, and so on. This measurement is the same as Measurement Example of a Bandpass Filter in the Quick Start.

Description of the Program

When you run this VBA program, reset is performed, the measurement conditions are automatically set, and the message "Perform the full 2-port calibration" is displayed. To perform the full 2-port calibration, click **Yes**; otherwise click **No**.

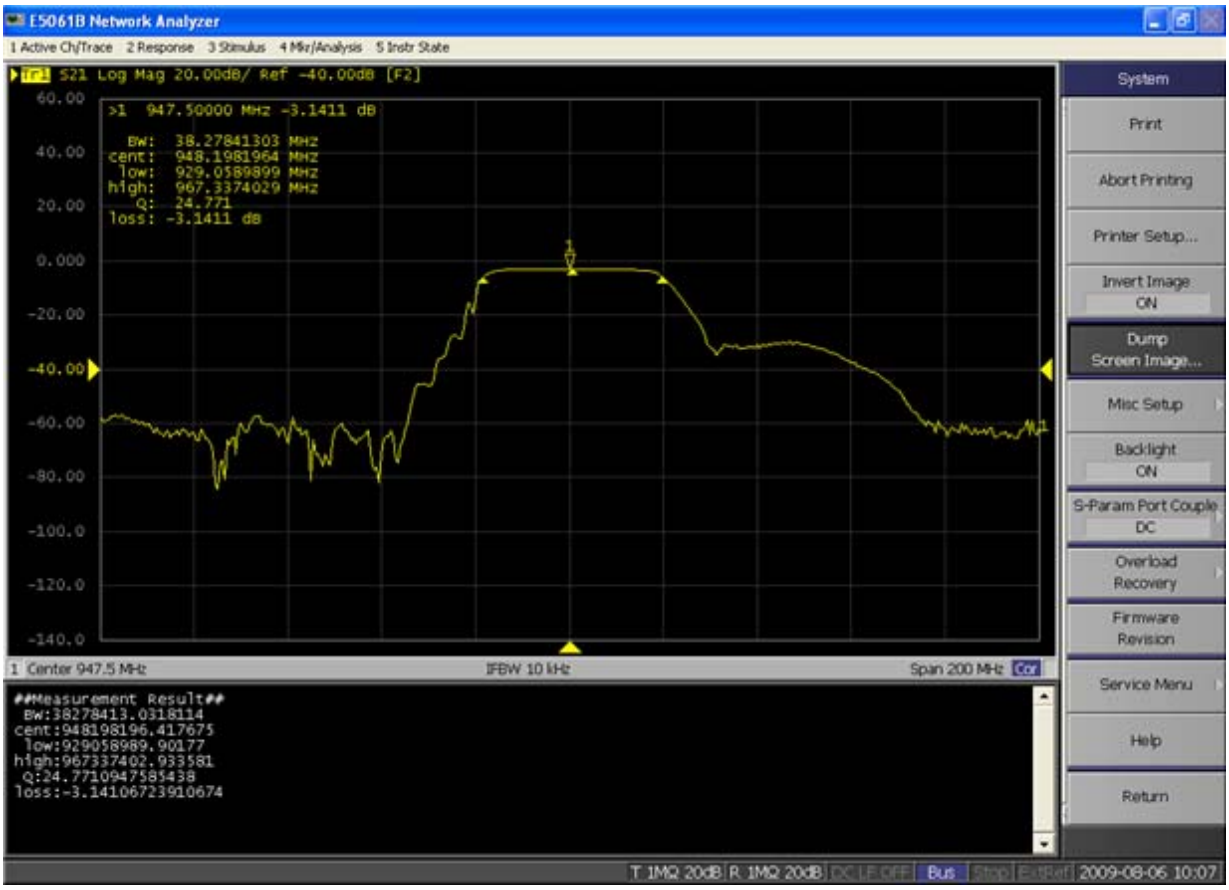
To perform the calibration, follow the onscreen messages to connect each standard of the Agilent 85032F calibration kit to the specified port, and then click **OK** to measure the calibration data. Click **Cancel** to return to the beginning of the calibration. You cannot skip the isolation calibration. When the calibration data measurement for all standards is complete, the message "All calibration data completion" is displayed, and the calibration coefficient is calculated.

NOTE When you cancel the calibration data measurement before completing the measurement of necessary calibration data, the settings condition may not be returned to its former state.

Then, the message "Connect DUT, and then press **Macro Setup** > **Continue**" is displayed in the instrument status bar in the lower part of the LCD display. Connect a DUT and perform **Macro Setup** > **Continue**. After the

measurement, the search result is displayed in the echo window, as shown below. If no bandwidth search target is found, only the result of the insertion loss obtained with the marker is displayed.

Example of display after executing the program in "apl_bsc.vba"



The basic measurement program (object name: mdlBscMeas) is described in detail below. Line numbers are added for description purpose only and do not appear in the actual program source code.

Measuring a band-pass filter (object name: mdlBscMeas)

```
""
' This VBA program is sample program to perform the basic measurement of the band-pass filter.
""
'
Sub Main()
    Dim Par As String, Fmt As String, File As String
    Dim Center As Double, Span As Double, IfBw As Double, Pow As Double
    Dim Bw As Double, Cent As Double
    Dim CutLow As Double, CutHigh As Double
```

```
    Dim Qfac As Double, Loss As Double
    Dim MkrVal As Variant, BwData As Variant
    Dim Nop As Long, NumTrac As Long, CalKit As Long, Buff As Long
    Dim Port As Variant, Error As Variant
'
' Store the sweep center value (947.5 MHz), the sweep span value (200 MHz), the number of measurement points (401),
' the IF bandwidth (10 kHz), and the power level (-10 dBm) into the variables Center, Span, Nop, IfBw, and Pow, respectively.
'
    Center = 947500000#
    Span = 200000000#
    Nop = 401
    IfBw = 10000#
    Pow = -10
'
' Store the number of traces (1), the measurement parameter (S21), the data format (log amplitude), the calibration kit
' number (4: 85032F), and the save file name (State08.sta) into the variables, NumTrac, Par, Fmt, CalKit, and File, respectively.
'
    NumTrac = 1
    Par = "S21"
    Fmt = "MLOG"
    CalKit = 4
    File = "State08.sta"
'
' Returns the E5061B to the preset state.
'
    SCPI.SYSTem.PRESet
'
' For channel 1, turn on the continuous trigger startup mode to On and set the trigger source to the bus trigger.
'
    SCPI.INITiate(1).CONTinuous = True
    SCPI.TRIGger.SEQuence.Source = "BUS"
'
' For channel 1, set the sweep center value to the Center variable, the sweep span value to the Span variable,
```

```
' the number of measurement points to the Nop variable, the IF bandwidth to the IfBw variable, and the power level
' to the Pow variable.
'
    SCPI.SENSe(1).FREQuency.Center = Center
    SCPI.SENSe(1).FREQuency.Span = Span
    SCPI.SENSe(1).SWEep.POINts = Nop
    SCPI.SENSe(1).BANDwidth.RESolution = IfBw
    SCPI.Source(1).POWer.LEVel.IMMediate.AMPLitude = Pow
'
' For channel 1, set the number of traces to the NumTrac variable, the measurement parameter to the Par variable,
' and the data format to the Fmt variable.
'
    SCPI.CALCulate(1).PARameter.Count = NumTrac
    SCPI.CALCulate(1).PARameter(1).DEFine = Par
    SCPI.CALCulate(1).PARameter(1).Select
    SCPI.CALCulate(1).SELected.Format = Fmt
'
' Stores the calibration kit number for channel 1 into the CalKit variable, and stores 1 and 2 into the Port variable
' that indicates ports used for the full 2-port calibration. Then, calls the Calib_Solt procedure.
'
    SCPI.SENSe(1).CORRection.COLLect.CKIT.Select = CalKit
    Port = Array(1, 2)
    Calib_Solt 1, 2, Port
'
' Save the instrument setting and the calibration coefficient into a file whose name is specified with the File variable.
'
    SCPI.MMEMory.STORe.STYPe = "CST"
    SCPI.MMEMory.STORe.STATe = File
'
' Displays a message that prompts you to connect a DUT (Device Under Test) in the instrument status bar in the lower part
' of the LCD display and waits for the operation of [Macro Setup > Continue] after the connection.
'
Meas_Start:
```

```
    Prompt ("Connect DUT, and then press [Macro Setup]-Continue button.")
'
' Generate a trigger to start a single sweep and wait until the measurement finishes (1 is read out with
' the SCPI.IEEE4882.OPC object).
'
    SCPI.TRIGger.SEQuence.SINGle
    Dmy = SCPI.IEEE4882.OPC
'
' For trace 1 of channel 1, executes auto scale to set the optimum scale.
'
    SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.AUTO
'
' Display marker 1 and move it so that the stimulus value becomes equal to the value of the Center
variable.
' Then, these lines read out the response value of marker 1 and store it into the MkrVal variable.
'
    SCPI.CALCulate(1).SELected.MARKer(1).STATe = True
    SCPI.CALCulate(1).SELected.MARKer(1).X = Center
    MkrVal = SCPI.CALCulate(1).SELected.MARKer(1).Y
'
' Enables the error handling routine starting from Bw_Err. If a runtime error occurs, the program goes
' to the error handling routine.
'
    On Error GoTo Bw_Err
'
' Set the bandwidth definition value to -3 dB and the bandwidth search result display to on, read out
' the bandwidth search result (bandwidth, center frequency, Q value, and insertion loss), and
' store it into the BwData variable.
'
    SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.THReshold = -3
    SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.STATe = True
    BwData = SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.DATA
'
' Based on the bandwidth search result, these lines store the bandwidth to the Bw variable, the center
frequency
' to the Cent variable, the Q value to the Qfac variable, and the insertion loss to the Loss variable.
' Then, the program goes to the processing starting from Skip_Bw_Err.
'
```

```
    Bw = BwData(0)
    Cent = BwData(1)
    Qfac = BwData(2)
    Loss = BwData(3)
    GoTo Skip_Bw_Err
'
' Define a runtime error handler. These lines read out and display the error number and error message of
the
' error that occurred and store 0 to the Bw, Cent, and Qfac variables and the response value of marker 1
' (MkrVal(0) variable) to the Loss variable. Then, the program finishes the error handling and
' proceeds to the next processing.
'
Bw_Err:
    Error = SCPI.SYSTem.Error
    MsgBox "Error No:" & Error(0) & " , Description:" & Error( 1)
    Bw = 0
    Cent = 0
    Qfac = 0
    Loss = MkrVal(0)
    Resume Skip_Bw_Err
'
' Calculate the 2 (higher and lower) cutoff frequencies from the values in the Bw and Cent variables
' and store them into the CutLow and CutHigh variables.
'
Skip_Bw_Err:
    CutLow = Cent - Bw / 2
    CutHigh = Cent + Bw / 2
'
' Display the search result (the values of the Bw, Cent, CutLow, CutHigh, Qfac, and Loss variables)
' in the echo window.
'
    ECHO "##Measurement Result##"
    ECHO " BW:" & Bw
    ECHO "cent:" & Cent
    ECHO " low:" & CutLow
    ECHO "high:" & CutHigh
    ECHO " Q:" & Qfac
```

```
    ECHO "loss:" & Loss
    SCPI.DISPlay.TABLe.TYPE = "ECHO"
    SCPI.DISPlay.TABLe.STATe = True
'
' Display the message asking whether you want to perform measurement again. Click Yes to return to
' the DUT connection section, otherwise click No to terminate the program.
'
    Buff = MsgBox("Do you make another measurement?", vbYesNo, "Bandpass fileter measurement")
    If Buff = vbYes Then
      GoTo Meas_Start
    End If
'
End Sub
'
'
''''
' The following code is Calib_Solt procedure.
''''
'
Private Sub Calib_Solt(Chan As Long, SoltType As Long, Port As Variant)
    Dim Dmy As Long, I As Long, J As Long, Buff As Long
'
' Display the message that prompts for the execution of the full n-port calibration (specified with the
SoltType variable).
' Click Cancel to cancel the calibration.
'
Cal_Start:
    Buff = MsgBox("Perform the full " & SoltType & "-port cali bration.", vbOKCancel, "Full" & SoltType & "-
port calibration")
    If Buff = vbCancel Then
      GoTo Cal_Skip
    End If
'
' Set the calibration type to the full n-port calibration for the port specified with the Port variable.
'
    Select Case SoltType
    Case 1
      SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT1 = Port(0)
```

```
    Case 2
      SCPI.SENSe(Chan).CORRection.COLLect.METHod.SOLT2 = Port
    End Select
'
' Processing loops according to the selected calibration type.
'
    For I = 1 To SoltType
'
' Display the message that prompts for connecting the open standard to the specified port. These lines
' start the measurement of the open calibration data initiated by clicking OK after the connection and
' wait for the completion of the measurement. Click Cancel to return to the beginning of the calibration.
'
      Buff = MsgBox("Connect the Open standard to Port " & CS tr(Port(I - 1)) & ".", vbOKCancel, "Full" &
SoltType & "-port calibration")
      If Buff = vbOK Then
        SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.OPEN = Port(I - 1)
        Dmy = SCPI.IEEE4882.OPC
      Else
        GoTo Cal_Start
      End If
'
' Display the message that prompts for connecting the short standard to the specified port. These lines
' start the measurement of the short calibration data initiated by clicking OK after the connection and
' wait for the completion of the measurement. Click Cancel to return to the beginning of the calibration.
'
      Buff = MsgBox("Connect the Short standard to Port " & CStr(Port(I - 1)) & ".", vbOKCancel, "Full" &
SoltType & "-port calibration")
      If Buff = vbOK Then
        SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.Short = Port(I - 1)
        Dmy = SCPI.IEEE4882.OPC
      Else
        GoTo Cal_Start
      End If
'
' Display the message that prompts for connecting the load standard to the specified port. These lines
' start the measurement of the load calibration data initiated by clicking OK after the connection and
' wait for the completion of the measurement. Click Cancel to return to the beginning of the calibration.
'
```

```
    Buff = MsgBox("Connect the Load standard to Port " & CStr(Port(I - 1)) & ".", vbOKCancel, "Full" &
SoltType & "-port calibration")
    If Buff = vbOK Then
      SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.Load = Port(I - 1)
      Dmy = SCPI.IEEE4882.OPC
    Else
      GoTo Cal_Start
    End If
'
  Next I
'
' Display the message that prompts for connecting the thru standard between the specified ports.
' These lines start the measurement of the thru calibration data initiated by clicking OK after the
' connection and wait for the completion of the measurement. Click Cancel to return to the
' beginning of the calibration.
'
  For I = 1 To SoltType - 1
    For J = I + 1 To SoltType
      Buff = MsgBox("Connect the Thru standard between Por t " & CStr(Port(I - 1)) & _
      " and Port " & CStr(Port(J - 1)) & ".", vbOKCancel, "Full" & SoltType & "-port calibration")
      If Buff = vbOK Then
        SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.THRU = Array(Port(I - 1 ), Port(J - 1))
        Dmy = SCPI.IEEE4882.OPC
        SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.THRU = Array(Port(J - 1 ), Port(I - 1))
        Dmy = SCPI.IEEE4882.OPC
      Else
        GoTo Cal_Start
      End If
    Next J
  Next I
'
' When the calibration type is not the 1-port calibration (a value other than 1 is specified for the
' SoltType variable), displays the message asking you whether you want to measure the isolation
' calibration data. When Yes is clicked, displays the message that prompts for connecting
' the load standard to the specified two ports (specified with the Port(I-1) and Port(J-1) variables).
' These lines start the measurement of the isolation calibration data initiated by clicking OK after the
' connection and wait for the completion of the measurement. Click Cancel to return to the
```

```
' beginning of the calibration.
'
   If SoltType <> 1 Then
     Buff = MsgBox("Do you measure the Isolation (Optional) ?", vbYesNo, "Full" & SoltType & "-port
calibration")
     If Buff = vbYes Then
       For I = 1 To SoltType - 1
         For J = I + 1 To SoltType
           Buff = MsgBox("Connect the Load standard to Port " & Port(I - 1) & " and Port " & Port(J - 1) & ".",
_
           vbOKCancel, "Full" & Solt Type & "-port calibration")
           If Buff = vbOK Then
             SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.ISOLation = Array(Port(I - 1), Port(J - 1))
             Dmy = SCPI.IEEE4882.OPC
             SCPI.SENSe(Chan).CORRection.COLLect.ACQuire.ISOLation = Array(Port(J - 1), Port(I - 1))
             Dmy = SCPI.IEEE4882.OPC
           Else
             GoTo Cal_Start
           End If
         Next J
       Next I
     End If
   End If
'
' Calculate the calibration coefficients from the measured calibration data and turn on the
' error correction function. Then, these lines display a calibration completion message.
'
   SCPI.SENSe(1).CORRection.COLLect.SAVE
   MsgBox "All calibration data completion."
'
Cal_Skip:
'
End Sub
```

Connecting Hard Disk of External PC (shared folder)

Other topics about Application Programs

Overview

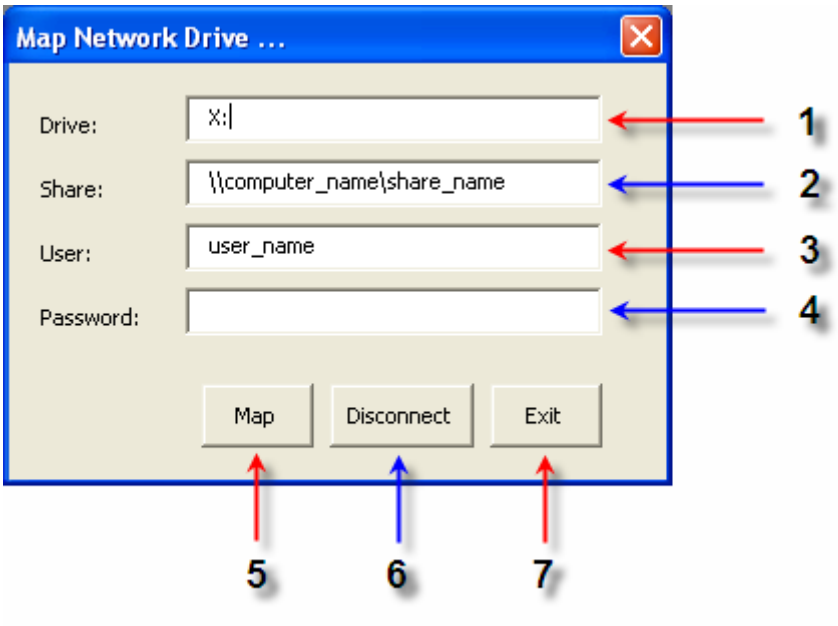
The **map_drive.vba** shows a sample program (VBA program) that demonstrates how to connect a hard disk (a shared folder) of an external PC to the E5061B. This VBA program consists of the following modules:

Object name	Module type	Description
frmMapDrive	User form	Connects or disconnects a hard disk
Module1	Standard module	Displays frmMapDrive

Using VBA Program

Load the **map_drive.vba** and press **Macro Run** key. The following macro appears.

Shared folder connection macro



Connecting (Mapping)

Enter the drive letter for the shared folder (1), share name of the shared folder (2), user name (3), password (4), and then click **Map** (5).

NOTE Consult your network administrator and enter the settings in the same way as done in the Windows PC operating system. If you enter an incorrect setting, an error might occur and the program might be interrupted.

Disconnecting

1. Enter the drive letter for the shared folder (1), and then click **Disconnect** (6).
2. Click **Exit** (7) to exit from the program.

Description of VBA Program

The program (object name: frmMapDrive) is described in detail below. The following description is included as a comment in the source code.

Sub CommandButton1_Click

This procedure is called when the user clicks the **Map** button. It checks whether the drive letter is used by using the IsDriveNameInUse procedure. Then the procedure connects the shared folder using the MapDrive procedure if the drive letter is not used or otherwise it displays a message to show the drive in use.

Sub CommandButton2_Click

This procedure is called when the user clicks the **Disconnect** button. The procedure disconnects the shared folder by using the DisconnectDrive procedure.

Function IsDriveNameInUse

This procedure checks if the txtDrive.Text (the drive letter specified by 1) is used.

Sub MapDrive

This procedure connects the shared folder as the txtDrive.Text (the drive letter specified by 1) drive by using the parameters: txtShare.Text (the share name specified by 2), txtUser.Text (the user name specified by 3), and txtPasswd.Text (the password specified by 4).

Sub DisconnectDrive

This procedure disconnects the txtDrive.Text (the drive letter specified by 1) drive.

Sub CommandButton3_Click

This procedure is called when the user clicks the **Exit** button. This procedure ends the program.

Connecting the hard disk of an external PC (Object name: frmMapDrive)

```
'
' This procedure is called when the user clicks the Map button. It checks whether
' the drive letter is used by using the IsDriveNameInUse procedure.
' Then the procedure connects the shared folder using the MapDrive procedure
' if the drive letter is not used or otherwise displays a message to show
' the drive letter is used.
'
Private Sub CommandButton1_Click()
  If Not IsDriveNameInUse Then
    Call MapDrive
  Else
    MsgBox "Drive """ & txtDrive.Text & """ is Already used", vb Critical
  End If
End Sub
'
' This procedure is called when the user clicks the Disconnect button.
' The procedure disconnects the shared folder by using the DisconnectDrive procedure.
'
Private Sub CommandButton2_Click()
  Call DisconnectDrive
End Sub
'
' This procedure checks if the txtDrive.Text is used.
'
Private Function IsDriveNameInUse() As Boolean
  Set fso = CreateObject("Scripting.FileSystemObject")
  IsDriveNameInUse = fso.DriveExists(txtDrive.Text)
End Function
'
' This procedure connects the shared folder as the txtDrive.Text drive by using the parameters:
' txtShare.Text, txtUser.Text, and txtPasswd.Text.
```

```
'
Private Sub MapDrive()
  Set network = CreateObject("wscript.network")
  Call network.MapNetworkDrive(txtDrive.Text, txtShare.Text, vbFal se, txtUser.Text, txtPasswd.Text)
End Sub
'
' This procedure disconnects the txtDrive.Text drive.
'
Private Sub DisconnectDrive()
  Set network = CreateObject("wscript.network")
  network.RemoveNetworkDrive txtDrive.Text
End Sub
'
' This procedure is called when the user clicks the Exit button. This procedure ends the program.
'
Private Sub CommandButton3_Click()
  Unload Me
End Sub
```

Complex Operation Library

Complex Operation Library

By using the complex operation library, you can perform operations of complex numbers.

Data of the complex type

In the complex operation library, you can use the complex type (Complex) as a data type. Data of the complex type consists of a real part (.real) and an imaginary part (.imag) as shown in the following example.

```
Dim Num as Complex
Num.real=1.0
Num.imag=2.0
```

List of procedures

The following table lists the procedures included in the complex operation library.

Procedure name	Function
ComplexSet(x,y)	Sets a complex number. (Specify a real part and an imaginary part.)
ComplexPolar(x,y)	Sets a complex number. (Specify an absolute value and a phase angle.)
ComplexSetArray(x)	Converts a variant type or double floating point type array to a complex type array.
ComplexAdd(x,y)	Returns the result of the addition.
ComplexSub(x,y)	Returns the result of the subtraction.
ComplexMul(x,y)	Returns the result of the multiplication.
ComplexDiv(x,y)	Returns the result of the division.
ComplexAbs(x)	Returns the absolute value.
ComplexArg(x)	Returns the phase angle.
ComplexNorm(x)	Returns the square of the absolute value.
ComplexConj(x)	Returns the conjugate complex number.
ComplexCos(x)	Returns the cosine.

ComplexCosh(x)	Returns the hyperbolic cosine.
ComplexSin(x)	Returns the sine.
ComplexSinh(x)	Returns the hyperbolic sine.
ComplexExp(x)	Returns e^x.
ComplexLog(x)	Returns the natural logarithm.
ComplexLog10(x)	Returns the common logarithm.
ComplexSqrt(x)	Returns the square root.

Sample Program

```
'
'     :
'     :
' The source code in this part is omitted.
'     :
'     :
'
  Dim Dmy As Long
  Dim s21_raw As Variant
  Dim s12_raw As Variant
  Dim s21_Comp As Complex
  Dim s12_Comp As Complex
  Dim trAce_ratio_comp As Complex
  Dim trAce_ratio(401) As Double
'
  SCPI.DISPlay.Split = "D1"
  SCPI.DISPlay.WINDow(1).Split = "D12_34"
  SCPI.CALCulate(1).PARameter.Count = 2
  SCPI.CALCulate(1).PARameter(1).DEFine = "s21"
  SCPI.CALCulate(1).PARameter(2).DEFine = "s12"
  SCPI.SENSe(1).SWEep.POINts = 201
'
  SCPI.TRIGger.SEQuence.Source = "bus"
  SCPI.TRIGger.SEQuence.SINGle
  Dmy = SCPI.IEEE4882.OPC
```

```
'
'** Get corrected data array of S21 and S12.
'
  SCPI.CALCulate(1).PARameter(1).SELect
  s21_raw = SCPI.CALCulate(1).SELected.DATA.SDATa
  SCPI.CALCulate(1).PARameter(2).SELect
  s12_raw = SCPI.CALCulate(1).SELected.DATA.SDATa
'
  For i = 0 To 200
'
'** Copy corrected data array to the complex data array
'** to take advantage of complex operation library
'
    s21_Comp = ComplexSet(s21_raw(2 * i), s21_raw(2 * i + 1))
    s12_Comp = ComplexSet(s12_raw(2 * i), s12_raw(2 * i + 1))
'
'** Calculate the ratio of S12 and S21
'** S12/S21
'
    trAce_ratio_comp = ComplexDiv(s12_Comp, s21_Comp)
    trAce_ratio(2 * i) = trAce_ratio_comp.real
    trAce_ratio(2 * i + 1) = trAce_ratio_comp.imag
'
  Next i
'
'
'
  SCPI.CALCulate(1).PARameter.Count = 4
'
'** Write "S12/S21" data to corrected data array for the trace 3 (LogMag)
'
  SCPI.CALCulate(1).PARameter(3).SELect
  SCPI.CALCulate(1).SELected.Format = "MLOG"
  SCPI.CALCulate(1).SELected.DATA.SDATa = trAce_ratio
'
'** Write "S12/S21" data to corrected data array for the trace 4 (Phase)
'
```

```
  SCPI.CALCulate(1).PARameter(4).SELect
  SCPI.CALCulate(1).SELected.Format = "PHASe"
  SCPI.CALCulate(1).SELected.DATA.SDATa = trAce_ratio
'
'     :
'     :
' The source code in this part is omitted.
'     :
'     :
'
```

Complex Operation Library

By using the complex operation library, you can perform operations of complex numbers.

Data of the complex type

In the complex operation library, you can use the complex type (Complex) as a data type. Data of the complex type consists of a real part (.real) and an imaginary part (.imag) as shown in the following example.

```
Dim Num as Complex
Num.real=1.0
Num.imag=2.0
```

List of procedures

The following table lists the procedures included in the complex operation library.

Procedure name	Function
ComplexSet(x,y)	Sets a complex number. (Specify a real part and an imaginary part.)
ComplexPolar(x,y)	Sets a complex number. (Specify an absolute value and a phase angle.)
ComplexSetArray(x)	Converts a variant type or double floating point type array to a complex type array.
ComplexAdd(x,y)	Returns the result of the addition.
ComplexSub(x,y)	Returns the result of the subtraction.
ComplexMul(x,y)	Returns the result of the multiplication.
ComplexDiv(x,y)	Returns the result of the division.
ComplexAbs(x)	Returns the absolute value.
ComplexArg(x)	Returns the phase angle.
ComplexNorm(x)	Returns the square of the absolute value.
ComplexConj(x)	Returns the conjugate complex number.
ComplexCos(x)	Returns the cosine.
ComplexCosh(x)	Returns the hyperbolic cosine.

ComplexSin(x)	Returns the sine.
ComplexSinh(x)	Returns the hyperbolic sine.
ComplexExp(x)	Returns e^x.
ComplexLog(x)	Returns the natural logarithm.
ComplexLog10(x)	Returns the common logarithm.
ComplexSqrt(x)	Returns the square root.

Sample Program

```
'
'     :
'     :
' The source code in this part is omitted.
'     :
'     :
'
  Dim Dmy As Long
  Dim s21_raw As Variant
  Dim s12_raw As Variant
  Dim s21_Comp As Complex
  Dim s12_Comp As Complex
  Dim trAce_ratio_comp As Complex
  Dim trAce_ratio(401) As Double
'
  SCPI.DISPlay.Split = "D1"
  SCPI.DISPlay.WINDow(1).Split = "D12_34"
  SCPI.CALCulate(1).PARameter.Count = 2
  SCPI.CALCulate(1).PARameter(1).DEFine = "s21"
  SCPI.CALCulate(1).PARameter(2).DEFine = "s12"
  SCPI.SENSe(1).SWEep.POINts = 201
'
  SCPI.TRIGger.SEQuence.Source = "bus"
  SCPI.TRIGger.SEQuence.SINGle
  Dmy = SCPI.IEEE4882.OPC
'
'** Get corrected data array of S21 and S12.
```

```
'
  SCPI.CALCulate(1).PARameter(1).SELect
  s21_raw = SCPI.CALCulate(1).SELected.DATA.SDATa
  SCPI.CALCulate(1).PARameter(2).SELect
  s12_raw = SCPI.CALCulate(1).SELected.DATA.SDATa
'
  For i = 0 To 200
'
'** Copy corrected data array to the complex data array
'** to take advantage of complex operation library
'
    s21_Comp = ComplexSet(s21_raw(2 * i), s21_raw(2 * i + 1))
    s12_Comp = ComplexSet(s12_raw(2 * i), s12_raw(2 * i + 1))
'
'** Calculate the ratio of S12 and S21
'** S12/S21
'
    trAce_ratio_comp = ComplexDiv(s12_Comp, s21_Comp)
    trAce_ratio(2 * i) = trAce_ratio_comp.real
    trAce_ratio(2 * i + 1) = trAce_ratio_comp.imag
'
  Next i
'
'
'
  SCPI.CALCulate(1).PARameter.Count = 4
'
'** Write "S12/S21" data to corrected data array for the trace 3 (LogMag)
'
  SCPI.CALCulate(1).PARameter(3).SELect
  SCPI.CALCulate(1).SELected.Format = "MLOG"
  SCPI.CALCulate(1).SELected.DATA.SDATa = trAce_ratio
'
'** Write "S12/S21" data to corrected data array for the trace 4 (Phase)
'
  SCPI.CALCulate(1).PARameter(4).SELect
  SCPI.CALCulate(1).SELected.Format = "PHASe"
```

```
  SCPI.CALCulate(1).SELected.DATA.SDATa = trAce_ratio
'
'      :
'      :
' The source code in this part is omitted.
'      :
'      :
'
```

Procedure Reference

ComplexAbs(*x*)

Syntax

Result = ComplexAbs(*x*)

Description

Returns the absolute value of complex number *x*.

Data type

x : Complex type (Complex)

Result : Double precision floating point type (Double)

Example of use

```
Dim a As Complex, b As Double
a = ComplexSet(1.5, 2.0)
b = ComplexAbs(a)
```

ComplexAdd(*x*,*y*)

Syntax

Result = ComplexAdd(*x*,*y*)

Description

Returns the result (x+y) of the addition to complex number *x* and another *y*.

Data type

x : Complex type (Complex)

y : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexAdd(a, b)
```

ComplexArg(*x*)

Syntax

Result = ComplexArg(*x*)

Description

Returns the phase angle (radian) of complex number *x*.

Data type

x : Complex type (Complex)

Result : Double precision floating point type (Double)

Example of use

```
Dim a As Complex, b As Double, c As Double, pi As Double
a = ComplexSet(1.5, 2.0)
b = ComplexArg(a)
pi = 3.14159265
c = b * 180 / pi      ‘ radian -> degree
```

ComplexConj(*x*)

Syntax

Result = ComplexConj(*x*)

Description

Returns the conjugate complex number of complex number *x*.

Data type

x : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexConj(a)
```

ComplexCos(*x*)

Syntax

Result = ComplexCos(*x*)

Description

Returns the cosine (cos(*x*)) of complex number *x*.

Data type

x : Complex type (Complex)
Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexCos(a)
```

ComplexCosh(*x*)

Syntax

Result = ComplexCosh(*x*)

Description

Returns the hyperbolic cosine (cosh(*x*)) of complex number *x*.

Data type

x : Complex type (Complex)
Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexCosh(a)
```

ComplexDiv(*x*,*y*)

Syntax

Result = ComplexDiv(*x*,*y*)

Description

Returns the result (x/y) of the division of complex number *x* and another *y*.

Data type

x : Complex type (Complex)

y : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexDiv(a, b)
```

ComplexExp(*x*)

Syntax

Result = ComplexExp(*x*)

Description

Returns the involution of the complex number *x*.

Data type

x : Complex type (Complex)
Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexExp(a)
```

ComplexLog(*x*)

Syntax

Result = ComplexLog(*x*)

Description

Returns the natural logarithm (log(*x*)) of complex number *x*.

Data type

x : Complex type (Complex)
Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexLog(a)
```

ComplexLog10(*x*)

Syntax

Result = ComplexLog(*x*)

Description

Returns the common logarithm (log 10(*x*)) of complex number *x*.

Data type

x : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexLog10(a)
```

ComplexMul(*x*,*y*)

Syntax

Result = ComplexMul(*x*,*y*)

Description

Returns the result (x\ y) of the multiplication of complex number *x* and another *y*.

Data type

x : Complex type (Complex)

y : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexMul(a, b)
```

ComplexNorm(*x*)

Syntax

Result = ComplexNorm(*x*)

Description

Returns the square of the absolute value of complex number *x*.

Data type

x : Complex type (Complex)

Result : Double precision floating point type (Double)

Example of use

```
Dim a As Complex, b As Double
a = ComplexSet(1.5, 2.0)
b = ComplexNorm(a)
```

ComplexPolar(*x*,*y*)

Syntax

z = ComplexPolar(*x*,*y*)

Description

Sets a complex number to a complex type variable *z*. Specifies a complex number with an absolute value *x* and a phase angle *y (radian)*.

Data type

x : Double precision floating point type (Double)

y : Double precision floating point type (Double)

z : Complex type (Complex)

Example of use

```
Dim a As Complex, pi As Double
pi = 3.14159265
a = ComplexPolar(2.5, 60 * pi / 180)
```

ComplexSet(*x*,*y*)

Syntax

z = ComplexSet(*x*,*y*)

Description

Sets a complex number to a complex type variable *z*. Specifies a complex number with a real part *x* and an imaginary part *y*. (Sets *x* and *y* to *z.real* and *z.imag* respectively.)

Data type

x : Double precision floating point type (Double)

y : Double precision floating point type (Double)

z : Complex type (Complex)

Example of use

```
Dim a as Complex
a = ComplexSet(1.5, 2.0)
```

ComplexSetArray(*x*)

Syntax

y = ComplexSetArray(*x*)

Description

Converts a variant type or double floating point type array *x* that contains complex numbers using 2 elements to store each complex number in the order of the real part and imaginary part to complex type array *y*.

Data type

x : Variant type (Variant) array or Double precision floating point type (Double) array

y : Complex type (Complex) array

Example of use

```
Dim a as Variant, b as Complex
a = SCPI.CALCulate(1).SELected.DATA.SDATa
b = ComplexSetArray(a)
```

ComplexSin(*x*)

Syntax

Result = ComplexSin(*x*)

Description

Returns the sine (sin(*x*)) of complex number *x*.

Data type

x : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSin(a)
```

ComplexSinh(*x*)

Syntax

Result = ComplexSinh(*x*)

Description

Returns the hyperbolic sine (sinh(*x*)) of complex number *x*.

Data type

x : Complex type (Complex)
Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSinh(a)
```

ComplexSqrt(*x*)

Syntax

Result = ComplexSqrt(*x*)

Description

Returns the square root ( ) of the complex number *x*.

Data type

x : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSqrt(a)
```

ComplexSub(*x*,*y*)

Syntax

Result = ComplexSub(*x*,*y*)

Description

Returns the result (x - y) of the subtraction complex number *x* and another *y*.

Data type

x : Complex type (Complex)

y : Complex type (Complex)

Result : Complex type (Complex)

Example of use

```
Dim a As Complex, b As Complex, c As Complex
a = ComplexSet(1.5, 2.0)
b = ComplexSet(0.5, 3.5)
c = ComplexSub(a, b)
```

Waveform Analysis Library

Waveform (Ripple) Analysis Library

Other topics about VBA Programming

Overview

By combining the COM objects provided with the E5061B and the ripple analysis library, you can easily perform the ripple analysis of waveforms.

Flow of Programming using Ripple Analysis Library

Below table shows the flow of the program development using the ripple analysis library. First, set up the analysis range and peak definition to use the procedures for ripple analysis.

1. Condition settings before using the ripple analysis library:
    - Specifying the analysis range
    - Setting the peak definition
2. Using the ripple analysis library

Condition Setting before using Ripple Analysis Library

Since the analysis conditions are not specified in the ripple analysis library, before using the procedure for ripple analysis, set up the analysis range and the peak definition using COM objects.

Specifying the Analysis Range

Use the following COM objects to specify the analysis range for ripple analysis.

- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STARt
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STOP
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle

Setting the Peak Definition

Use the following COM objects to set up the peak definition for ripple analysis.

- SCPI.CALCulate(Ch).SELected.FUNCtion.PEXCursion

- SCPI.CALCulate(Ch).SELected.FUNCtion.PPOLarity

List of Ripple Analysis Library

Use the provided procedures for ripple analysis to analyze the ripple of waveforms and output the result. All procedures perform analysis only within the stimulus range for the specified channel.

Function Name	Description
MaxPeakToPeak(Chan)	Returns the maximum value of the difference between a positive peak and a negative peak
MaxRightGap(Chan)	Returns the maximum value of the difference between a positive peak and its right adjacent negative peak.
MaxLeftGap(Chan)	Returns the maximum value of the difference between a positive peak and its left adjacent negative peak.
MaxGap(Chan)	Returns the maximum value of the difference between a positive peak and its adjacent negative peak.
MaxEnvelopeGap(Chan)	Returns the maximum value of the vertical distance between a line segment connecting 2 adjacent positive peaks and the negative peak between them.
GapMean(Chan)	Returns the mean value of the differences between a negative peak and its right and left adjacent positive peaks.
MaxRippleValue(Chan)	Returns the maximum value of the total of the differences between a negative peak and its right and left adjacent positive peaks.
MaxRipplePoint(Chan,Stim)	Returns the maximum value of the total of the differences between a negative peak and its right and left adjacent positive peaks and the

	stimulus value (*Stim*) of the valley of the ripple.
Pole(Chan,D,LeftStim,LeftValue, RightStim,RightValue)	Returns the values (*LeftValue* and *RightValue*) and the stimulus values (*LeftStimulus* and *RightStimulus*) of the right and left negative peaks detected first below the specified value (*D*) relative to the maximum value.
FirstRightGap(Chan)	Returns the difference between the positive peak detected first when searching from the left edge toward the right edge and its right adjacent negative peak.
FirstLeftGap(Chan)	Returns the difference between the positive peak detected first when searching from the right edge toward the left edge and its left adjacent negative peak.
FirstRightInterval(Chan)	Returns the difference of the stimulus value between the positive peak detected first when searching from the left edge toward the right edge and its right adjacent negative peak.
FirstLeftInterval(Chan)	Returns the difference of the stimulus value between the positive peak detected first when searching from the left edge toward the right edge and its left adjacent negative peak.

Sample Program

Here is a simple sample program using the ripple analysis procedures.

```
Sub Sample()

Dim Val As Double    (1)

SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion = 1.5    (2)
SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity = "BOTH"    (2)
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt = 935E6    (3)
```

```
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 960E6    (3)
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe = True    (3)
  .
  .
Val = MaxPeakToPeak(1)     (4)

End Sub
```

Let us break down the code into a number of blocks and see what they do.

1. Defines a variable Val as Double.
2. Sets the lower limit of the peak excursion value and polarity of the peak search to 1.5 and both positive peak and negative peak, respectively.
3. Sets the analysis range of channel 1 to 935 MHz to 960 MHz.
4. For channel 1, substitute the return value from the MaxPeakToPeak function (procedure) in the ripple analysis library to the Val variable.

Waveform (Ripple) Analysis Library

Other topics about VBA Programming

Overview

By combining the COM objects provided with the E5061B and the ripple analysis library, you can easily perform the ripple analysis of waveforms.

Flow of Programming using Ripple Analysis Library

Below table shows the flow of the program development using the ripple analysis library. First, set up the analysis range and peak definition to use the procedures for ripple analysis.

1. Condition settings before using the ripple analysis library:
    - Specifying the analysis range
    - Setting the peak definition
2. Using the ripple analysis library

Condition Setting before using Ripple Analysis Library

Since the analysis conditions are not specified in the ripple analysis library, before using the procedure for ripple analysis, set up the analysis range and the peak definition using COM objects.

Specifying the Analysis Range

Use the following COM objects to specify the analysis range for ripple analysis.

- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STARt
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STOP
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe
- SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle

Setting the Peak Definition

Use the following COM objects to set up the peak definition for ripple analysis.

- SCPI.CALCulate(Ch).SELected.FUNCtion.PEXCursion
- SCPI.CALCulate(Ch).SELected.FUNCtion.PPOLarity

List of Ripple Analysis Library

Use the provided procedures for ripple analysis to analyze the ripple of waveforms and output the result. All procedures perform analysis only within the stimulus range for the specified channel.

Function Name	Description
MaxPeakToPeak(Chan)	Returns the maximum value of the difference between a positive peak and a negative peak
MaxRightGap(Chan)	Returns the maximum value of the difference between a positive peak and its right adjacent negative peak.
MaxLeftGap(Chan)	Returns the maximum value of the difference between a positive peak and its left adjacent negative peak.
MaxGap(Chan)	Returns the maximum value of the difference between a positive peak and its adjacent negative peak.
MaxEnvelopeGap(Chan)	Returns the maximum value of the vertical distance between a line segment connecting 2 adjacent positive peaks and the negative peak between them.
GapMean(Chan)	Returns the mean value of the differences between a negative peak and its right and left adjacent positive peaks.
MaxRippleValue(Chan)	Returns the maximum value of the total of the differences between a negative peak and its right and left adjacent positive peaks.
MaxRipplePoint(Chan,Stim)	Returns the maximum value of the total of the differences between a negative peak and its right and left adjacent positive peaks and the stimulus value (*Stim*) of the valley

	of the ripple.
Pole(Chan,D,LeftStim,LeftValue, RightStim,RightValue)	Returns the values (*LeftValue* and *RightValue*) and the stimulus values (*LeftStimulus* and *RightStimulus*) of the right and left negative peaks detected first below the specified value (*D*) relative to the maximum value.
FirstRightGap(Chan)	Returns the difference between the positive peak detected first when searching from the left edge toward the right edge and its right adjacent negative peak.
FirstLeftGap(Chan)	Returns the difference between the positive peak detected first when searching from the right edge toward the left edge and its left adjacent negative peak.
FirstRightInterval(Chan)	Returns the difference of the stimulus value between the positive peak detected first when searching from the left edge toward the right edge and its right adjacent negative peak.
FirstLeftInterval(Chan)	Returns the difference of the stimulus value between the positive peak detected first when searching from the left edge toward the right edge and its left adjacent negative peak.

Sample Program

Here is a simple sample program using the ripple analysis procedures.

```
Sub Sample()

Dim Val As Double    (1)

SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion = 1.5    (2)
SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity = "BOTH"    (2)
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt = 935E6    (3)
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 960E6    (3)
```

```
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe = True    (3)
  .
  .
Val = MaxPeakToPeak(1)     (4)

End Sub
```

Let us break down the code into a number of blocks and see what they do.

1. Defines a variable Val as Double.
2. Sets the lower limit of the peak excursion value and polarity of the peak search to 1.5 and both positive peak and negative peak, respectively.
3. Sets the analysis range of channel 1 to 935 MHz to 960 MHz.
4. For channel 1, substitute the return value from the MaxPeakToPeak function (procedure) in the ripple analysis library to the Val variable.

Procedure Reference

FirstLeftGap(*Chan*)

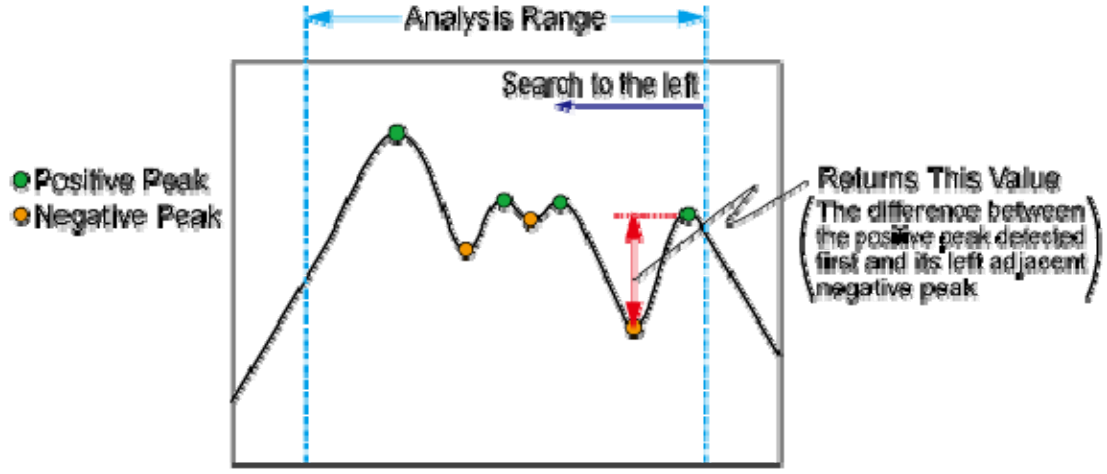
Syntax

Value = FirstLeftGap(*Chan*)

Description

Returns the response difference between the positive peak detected first when searched from the right edge toward the left edge within the analysis range and its left adjacent negative peak.

FirstLeftGap



e5071c431

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the response difference between the fist detected positive peak when searching from the right edge toward the left edge within the analysis range and its left adjacent negative peak.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double

Value = FirstLeftGap(1)
MsgBox "First Left Gap =" & Value
```

FirstLeftInterval(*Chan*)

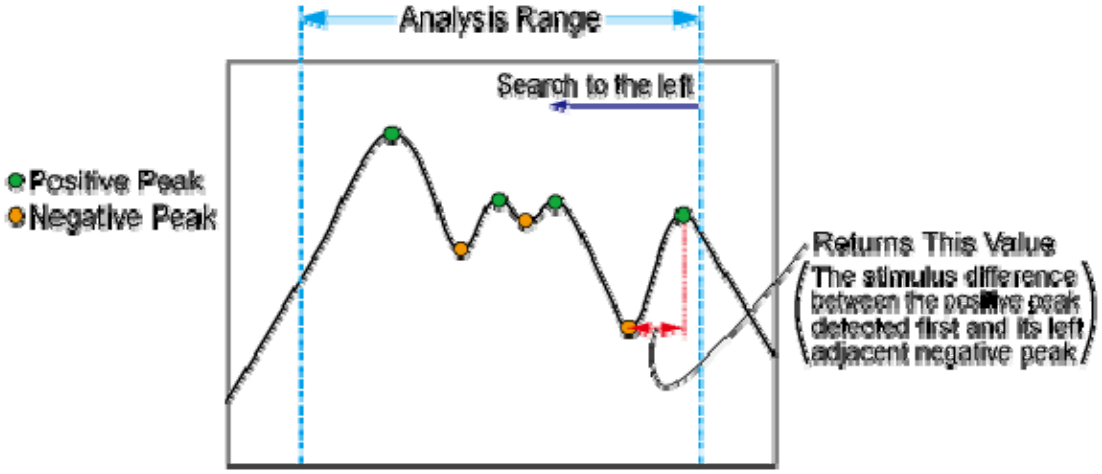
Syntax

Value = FirstLeftInterval(*Chan*)

Description

Returns the stimulus difference between the first detected positive peak when searching from the right edge toward the left edge within the analysis range and its left adjacent negative peak.

FirstLeftInterval



e5071c432

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the stimulus difference between the positive peak detected first when searched from the right edge toward the left edge within the analysis range and its left adjacent negative peak.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double

Value = FirstLeftInterval(1)
MsgBox "First Left Interval =" & Value
```

FirstRightGap(*Chan*)

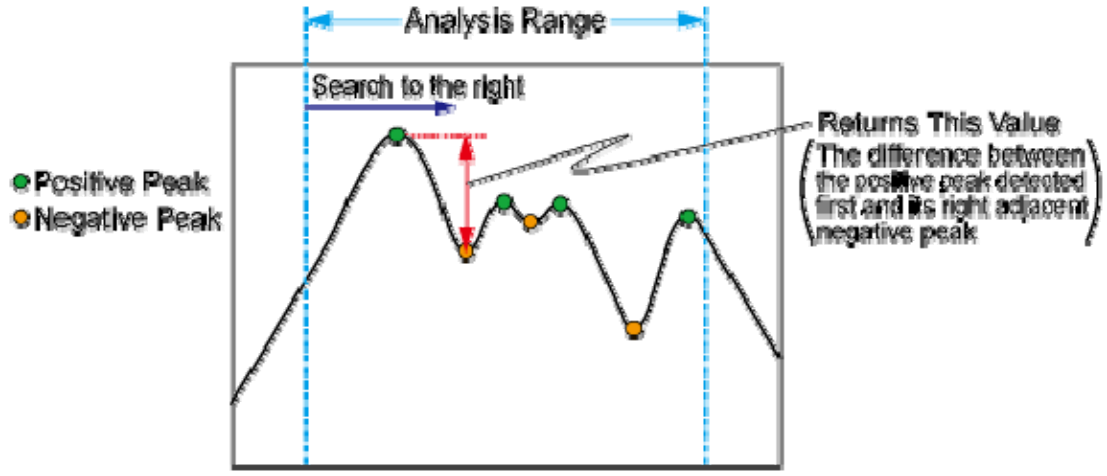
Syntax

Value = FirstRightGap(*Chan*)

Description

Returns the response difference between the fist detected positive peak when searching from the left edge toward the right edge within the analysis range and its right adjacent negative peak.

FirstRightGap



e5071c434

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the response difference between the first detected positive peak when searching from the left edge toward the right edge within the analysis range and its right adjacent negative peak.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Value = FirstRightGap(1)
MsgBox "First Right Gap =" & Value
```

FirstRightInterval(*Chan*)

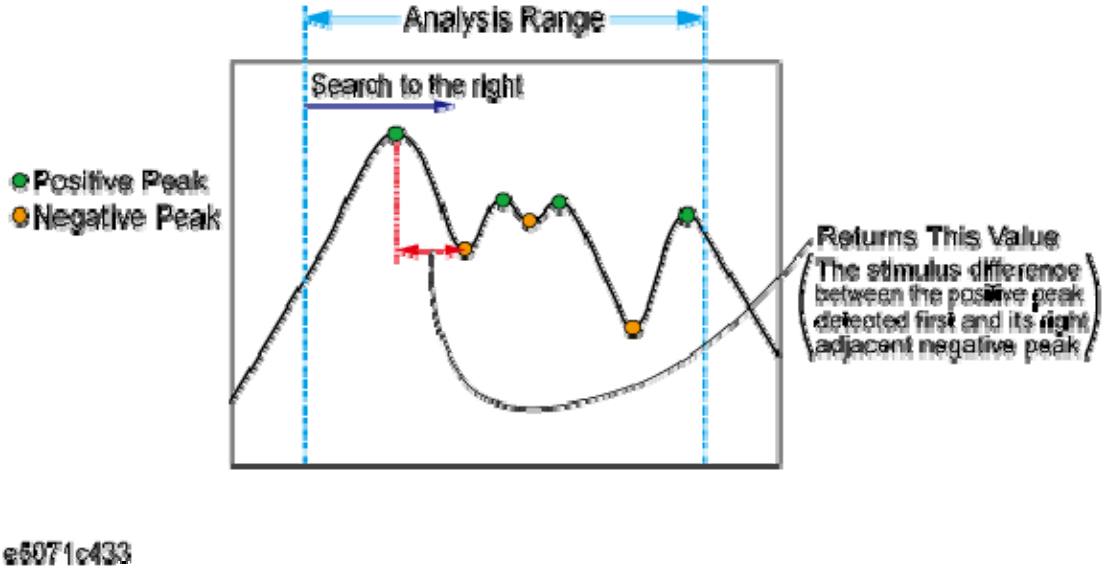
Syntax

Value = FirstRightInterval(*Chan*)

Description

Returns the stimulus difference between the positive peak detected first when searched from the left edge toward the right edge within the analysis range and its right adjacent negative peak.

FirstRightInterval



e5071c433

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the stimulus difference between the positive peak detected first when searched from the left edge toward the right edge within the analysis range and its right adjacent negative peak.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double

Value = FirstRightInterval(1)
MsgBox "First Right Interval =" & Value
```

GapMean(*Chan*)

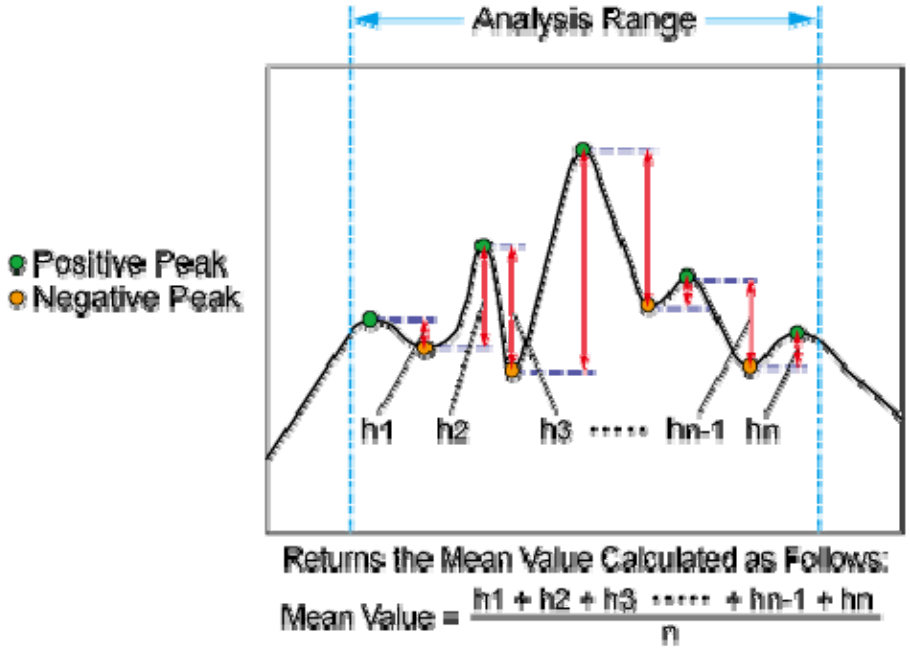
Syntax

Value = GapMean(*Chan*)

Description

Returns the mean value of the response differences between the negative peaks and its adjacent positive peaks within the analysis range.

GapMean



e5071c427

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9

Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the mean value of the response differences between the negative peaks and its right and left adjacent positive peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Value = GapMean(1)
MsgBox "Gap Mean =" & Value
```

MaxEnvelopeGap(*Chan*)

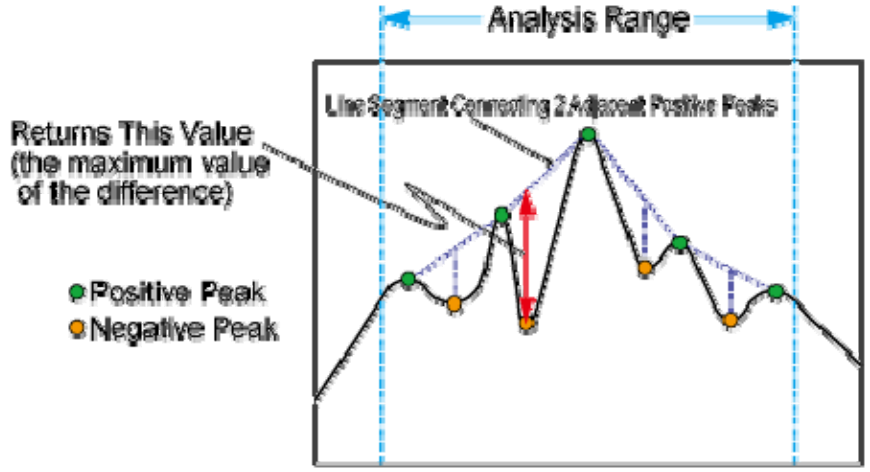
Syntax

Value = MaxEnvelopeGap(*Chan*)

Description

Returns the maximum value of the vertical distance between the line segments connecting 2 adjacent positive peaks and the negative peaks between them within the analysis range.

MaxEnvelopeGap



e5071c426

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the maximum value of the vertical distance between the line segments connecting 2 adjacent positive peaks and the negative peaks between them.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Value = MaxEnvelopeGap(1)
MsgBox "Max Envelope Gap =" & Value
```

MaxGap(*Chan*)

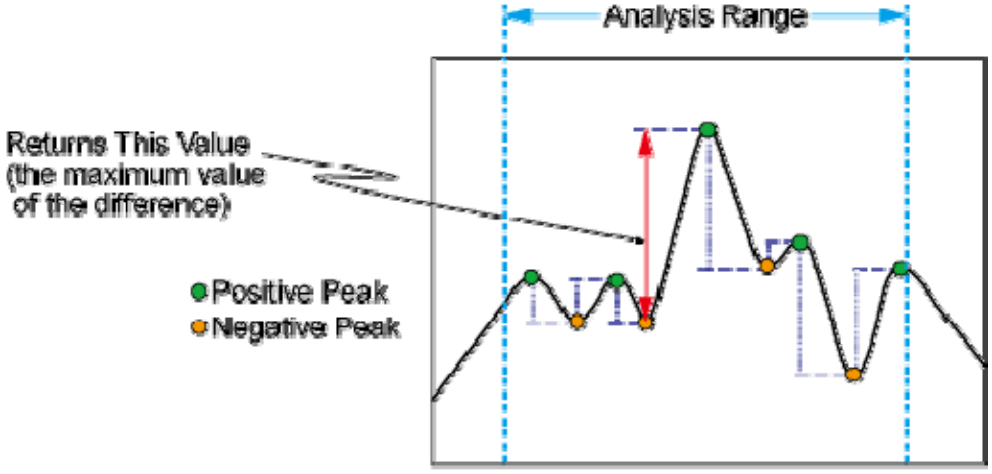
Syntax

Value = MaxGap(*Chan*)

Description

Returns the maximum value of the response differences between the positive peaks and its adjacent negative peaks within the analysis range.

MaxGap



e5071c425

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the maximum value of the response differences between the positive peaks and its adjacent negative peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Value = MaxGap(1)
MsgBox "Max Gap =" & Value
```

MaxLeftGap(*Chan*)

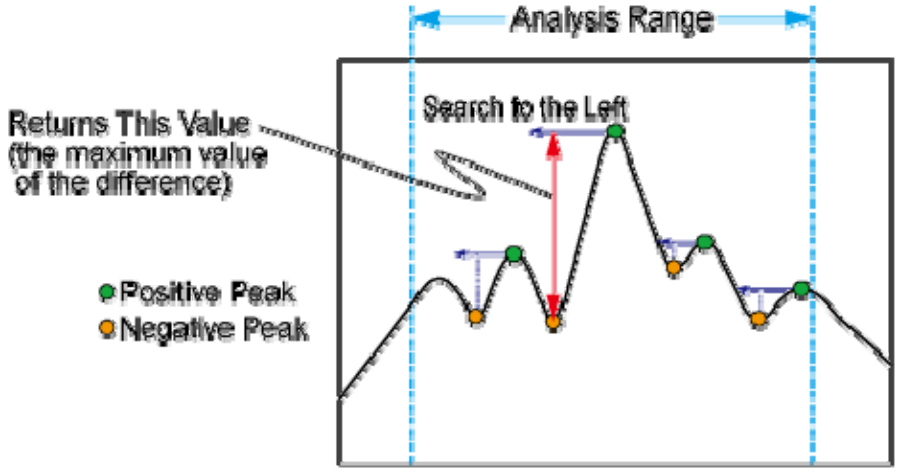
Syntax

Value = MaxLeftGap(*Chan*)

Description

Returns the maximum value of the response differences between the positive peaks and its left adjacent negative peaks within the analysis range.

MaxLeftGap



e5071c424

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the maximum value of the response differences between the positive peaks and its left adjacent negative peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Value = MaxLeftGap(1)
MsgBox "Max Left Gap =" & Value
```

MaxPeakToPeak(*Chan*)

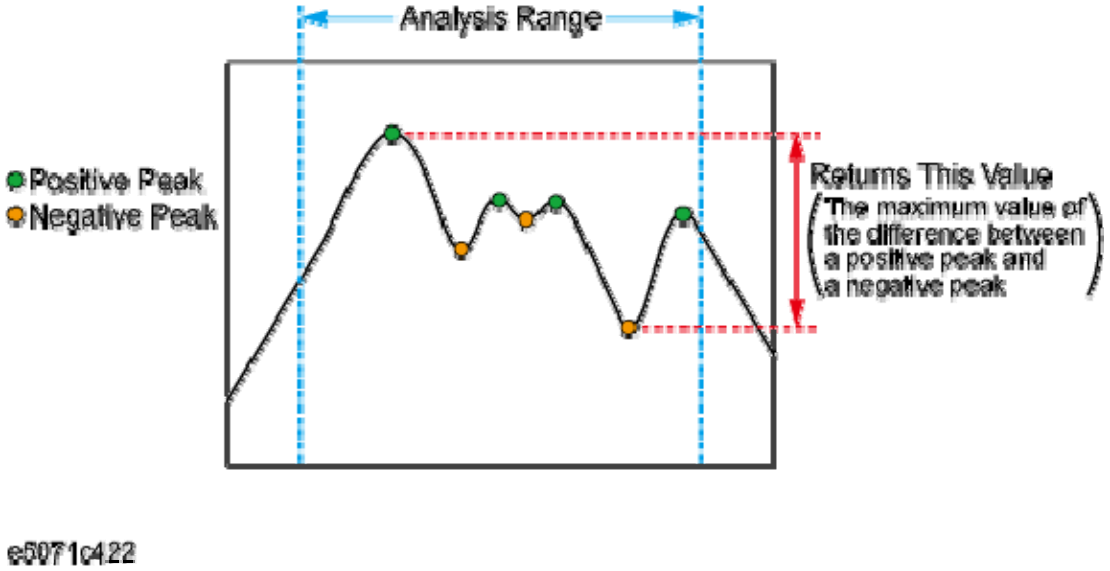
Syntax

Value = MaxPeakToPeak(*Chan*)

Description

Returns the maximum value of the response differences between the positive peaks and the negative peaks within the analysis range.

MaxPeakToPeak



Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the maximum value of the response differences between the positive peaks and the negative peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Value = MaxPeakToPeak(1)
MsgBox "Max Peak To Peak =" & Value
```

MaxRightGap(*Chan*)

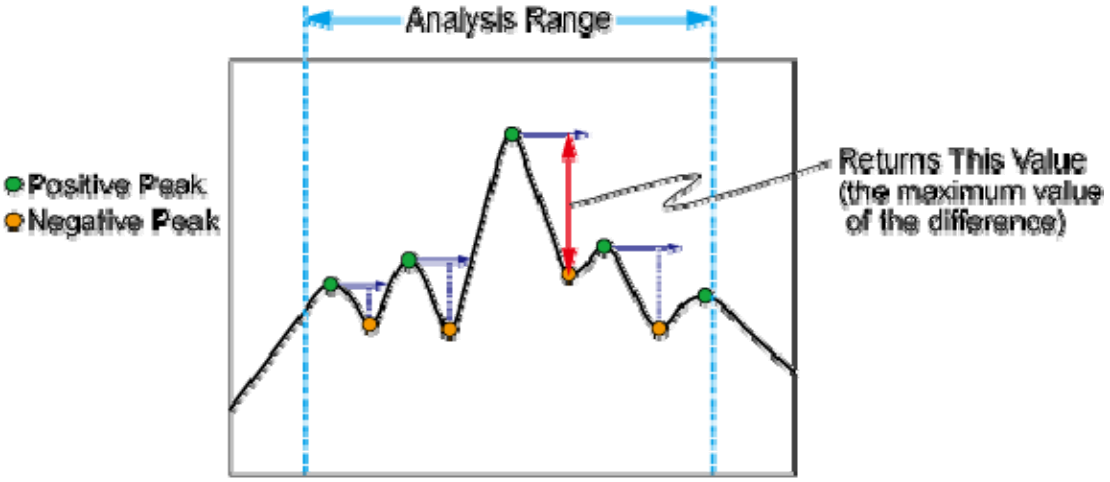
Syntax

Value = MaxRightGap(*chan*)

Description

Returns the maximum value of the response differences between the positive peaks and its right adjacent negative peaks within the analysis range.

MaxRightGap



e5071c423

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

998

Parameter	*Value*
Description	Returns the maximum value of the response differences between the positive peaks and its right adjacent negative peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double

Value = MaxRightGap(1)
MsgBox "Max Right Gap =" & Value
```

MaxRipplePoint(*Chan*,*Stim*)

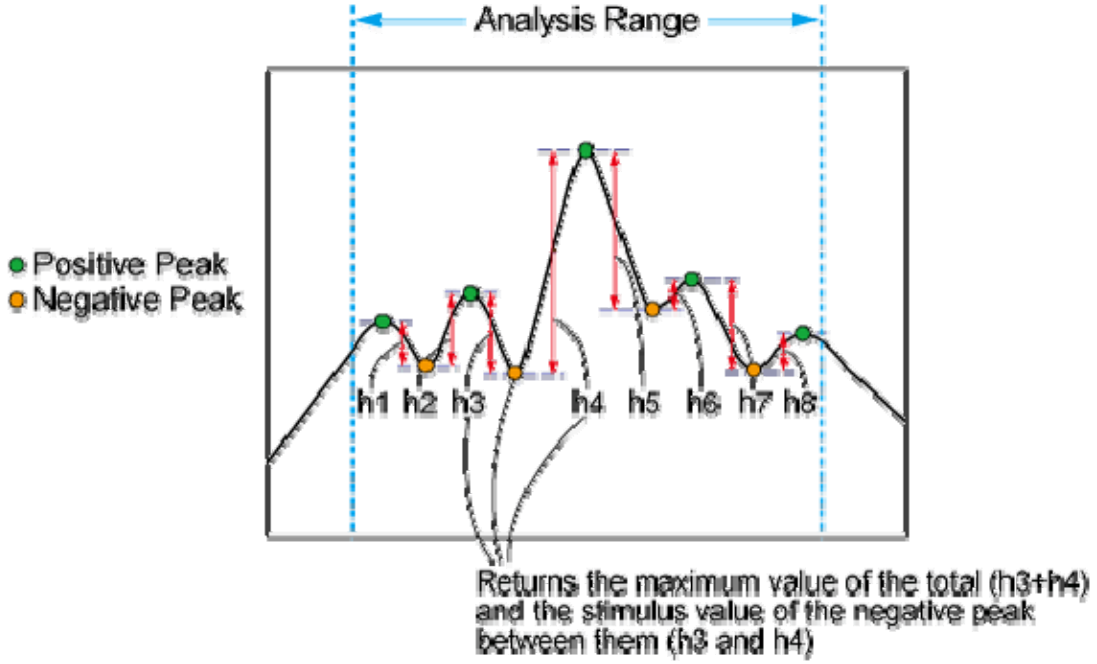
Syntax

Value = MaxRipplePoint(*Chan,Stim*)

Description

Returns the maximum value of the sum of the response differences between the negative peaks and its adjacent positive peaks and the stimulus value of the applicable negative peaks within the analysis range.

MaxRipplePoint



e5071c428

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9

Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the maximum value of the sum of the response differences between the negative peaks and its adjacent positive peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Parameter	*Stim*
Description	Returns the stimulus value of the negative peak at which the sum of the response differences between the negative peak and its adjacent positive peaks is maximum.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Dim Stim As Double

Value = MaxRipplePoint(1, Stim)
MsgBox "Max Ripple Value =" & Value & " , Stimulus =" & Stim
```

MaxRippleValue(*Chan*)

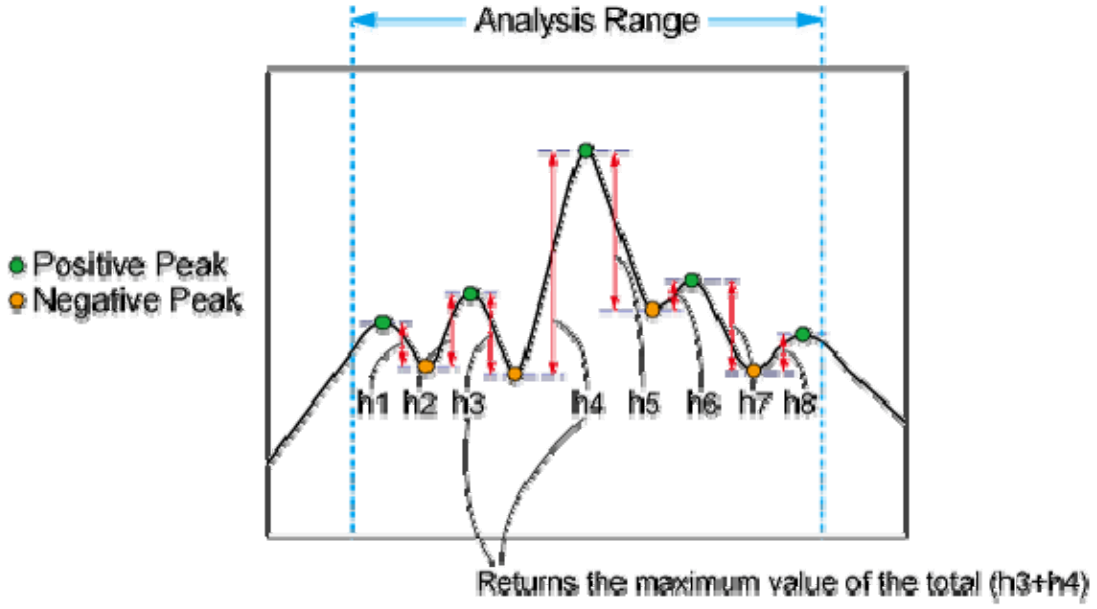
Syntax

Value = MaxRippleValue(*Chan*)

Description

Returns the maximum value of the sum of the response differences between the negative peaks and its adjacent positive peaks within the analysis range.

MaxRippleValue



e5071c429

Variable

Parameter	*Chan*
Description	Specifies the channel number.
Data type	Integer type (Integer)
Range	1 to 9

Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Return value

Parameter	*Value*
Description	Returns the maximum value of the sum of the response differences between the negative peaks and its adjacent positive peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim Value As Double
Value = MaxRippleValue(1)
MsgBox "Max Ripple Value =" & Value
```

Pole(*Chan*,*D*,*LeftStim*,*LeftValue*,*RightStim*,*RightValue*)

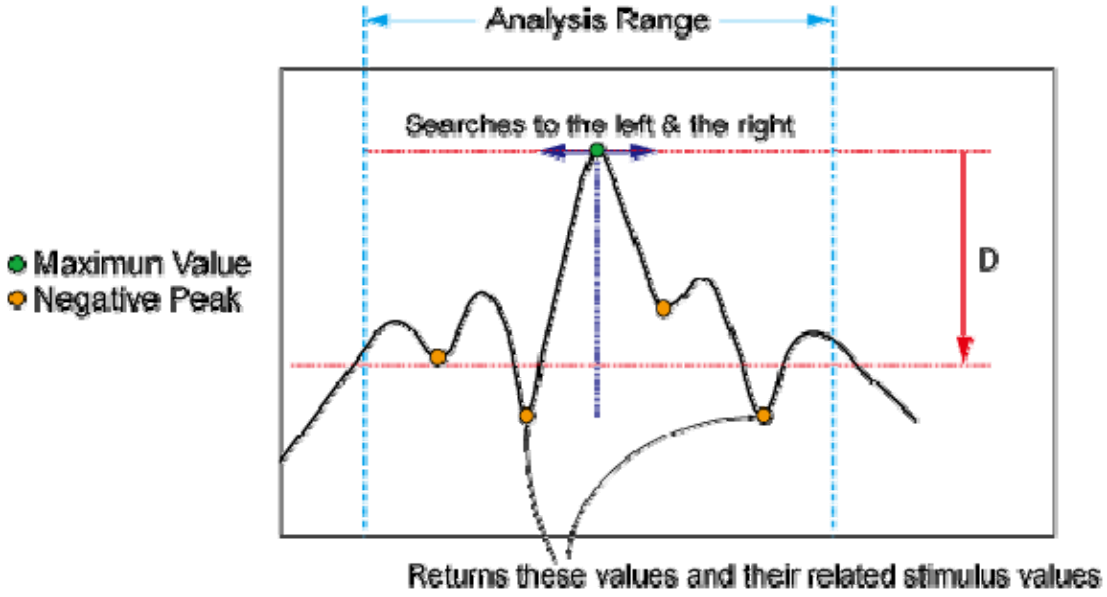
Syntax

Call Pole(*Chan*,*D*,*LeftStim*,*LeftValue*,*RightStim*,*RightValue*)

Description

For the negative peaks below the specified value (*D*) relative to the maximum value of the positive peaks within the analysis range, the response value (*LeftValue*) and stimulus value (*LeftStimulus*) of the negative peak first detected when searching to the left from the maximum value of the positive peaks, and the response value (*RightValue*) and stimulus value (*RightStimulus*) of the first detected negative peak when searching to the right from the maximum value of the positive peaks are returned.

Pole



e5071c430

Variable

Parameter	*Chan*
Description	Specifies the channel number.

Data type	Integer type (Integer)
Range	1 to 9
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Parameter	*D*
Description	Specifies the difference from the maximum value.
Data type	Double precision floating point type (Double)

Return value (arguments)

Parameter	*LeftStim*
Description	Returns the stimulus value of the first detected negative peak to the left from the maximum value of the positive peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Parameter	*LeftValue*

Description	Returns the response value of the first detected negative peak to the left from the maximum value of the positive peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Parameter	*RightStim*
Description	Returns the stimulus value of the first detected negative peak to the right from the maximum value of the positive peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Parameter	*RightValue*
Description	Returns the response value of the first detected negative peak to the right from the maximum value of the positive peaks.
Data type	Double precision floating point type (Double)
Note	If no applicable point is detected, 0 is returned.

Example of use

```
Dim LeftStim As Double
Dim LeftValue As Double
Dim RightStim As Double
```

```
Dim RightValue As Double

Call Pole(1, 1, LeftStim, LeftValue, RightStim, RightValue)

MsgBox "Left Pole =" & LeftStim & ":" & LeftValue
MsgBox "Right Pole =" & RightStim & ":" & RightValue
```

Command Reference

Notational Conventions

This section describes the notational conventions used for the description of commands reference.

Object Type

Object type describes different types of E5061B COM objects. The E5061B provides properties and methods as COM objects. COM objects which set (send)/read (return) the state of the E5061B using variables are defined as property and COM objects which does other processing are defined as method.

COM objects which are only used to read the state of the E5061B are indicated with "**Read-only**" and the ones used only to set the state of the E5061B are indicated by "**Write-only**" . COM object that can both read and write data to the E5061B are indicated by '**Read-Write**'.

Syntax

Syntax describes the syntax for sending a COM object from the E5061B VBA to the E5061B. The syntax consists of two parts: the object part and the set/read part, with an equal "=" inserted between them. Variables are indicated by italicized letters. Variables with () are indices. For indices with () having their preset values, you can omit "(*variable*)," and, if omitted, the preset values are automatically set.

There are some commands of which indices are not used. For example ,SCPI.CALCulate(Ch).PARameter.COUNt has no index for PARameter in the manual. However, PARameter can have an index like SCPI.CALCulate(Ch).PARameter(Tr).COUNt. In this case, specifying the trace number in brackets (Tr) has no meaning (same operation for any number).

The following table describes the 3 types of syntax for coding using objects:

Type	Description
"Object (property) = *variable*":	Set the stat of the E5061B.
variable=object (property):	Read the stat of the E5061B.
"Object (method)":	Perform some processing in the E5061B.

Description

Description describes how to use the COM object or the operation when executed.

Variable

Variable provides description about different variables that can be used with the COM objects. It gives the description, data type, allowable range, preset value, unit, resolution, and notes for *variable (italic)* shown in the syntax.

NOTE Variables declared as the string data type (String) are not case-sensitive. For variables of the string type that indicate arguments (written as *Param* in the syntax), you can omit lower-case letters.

The data types of the E5061B COM objects include 5 types as shown in the following table. Before using variables, declare the data type of each variable. If you do not declare the data type of a variable, it is automatically processed as a variant type.

Data type	Name	Consumed memory	Range
Long	Long integer type	4 bytes	-2,147,483,648 to 2,147,483,647 (-2^31 to 2^31 - 1)
Double	Double precision floating point type	8 bytes	For a negative value: -1.797693134862231E+308 to -4.940656458412465E-324 For a positive value: 4.940656458412465E-324 to 1.797693134862231E+308
Boolean	Boolean type	2 bytes	For COM: True or False (For SCPI: ON or OFF)
String	Character string type	1 byte / alphanumeric character	Up to approximately 2 billion characters
Variant	Variant type	16 bytes	No limitation
Binary	Byte		

Examples

Examples provide a sample of using the object through coding with the E5061B VBA.

Related Objects

Related objects provide information about other objects that are similar/related with the object.

Equivalent Key

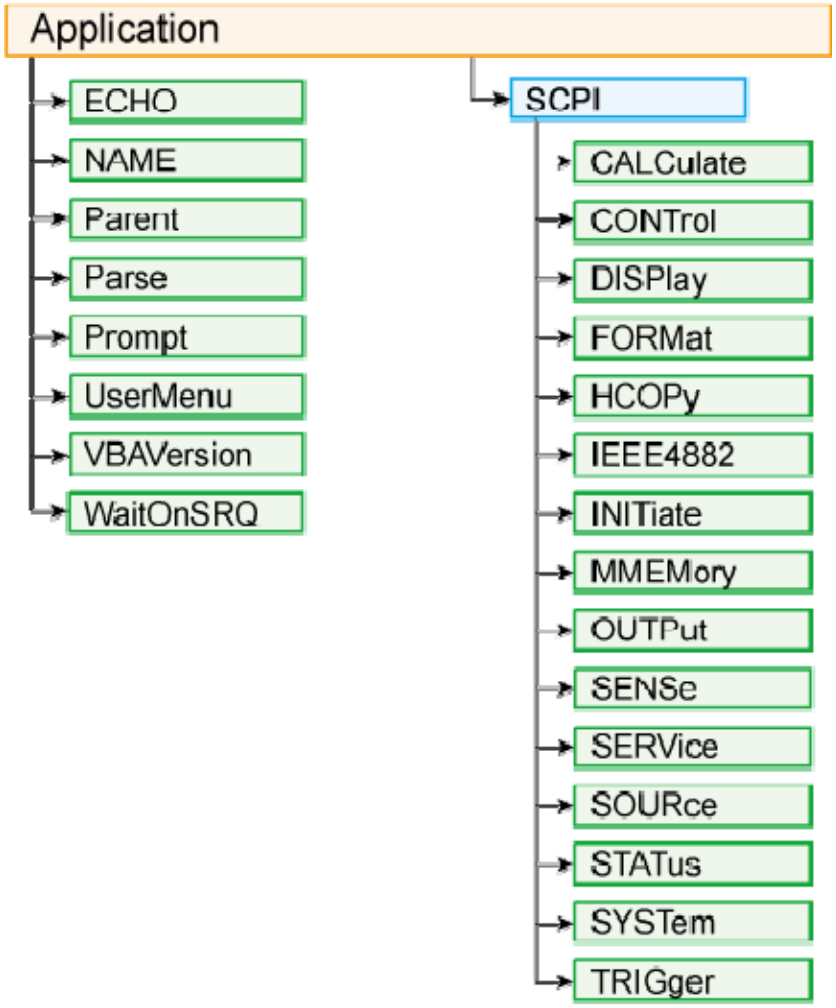
Equivalent key shows the operational procedure of the front panel keys that has the same effect as this object.

Equivalent SCPI command

Equivalent SCPI command shows the SCPI command to execute from an external controller. Its syntax, query response, and example of use are provided.

COM Object Model

The COM objects provided for the E5061B are structured hierarchically as shown below.



e5071c473

Application Objects

The Application objects are at the top of the hierarchy of the E5061B COM object model. They consist of 7 objects dedicated to the COM interface and SCPI objects corresponding to SCPI commands.

SCPI Objects

The SCPI objects are created to realize the SCPI commands of the E5061B with the COM interface.

In writing SCPI object messages, the conversion rules from the SCPI commands are as follows:

- SCPI. must be at the beginning. Notice that the IEEE common commands start with SCPI.IEEE4882. and "*" is omitted.
- Replace colons (:) used as the hierarchical separator symbol with dots (.).
- The number written in the object message is specified with ( ).
- You cannot omit the command message in the syntax.

SCPI command	COM object
OUTPUT 717;":SOUR1:POW -10"	SCPI.SOURce(1).POWer.LEVel.IMMediate.AMPLitude = -10
OUTPUT 717;":SENS1:CORR:COLL:METH:TYPE?" ENTER 717;A$	A = SCPI.SENSe(1).CORRection.COLLect.METHod:TYPE
OUTPUT 717;"*CLS"	SCPI.IEEE4882.CLS

Application Objects

ECHO

Object type

Method (**Write-only**)

Syntax

ECHO *V1,V2, ... ,V10*

ECHO *SCPI object*

Description

Displays in the echo window. (No read)

This command is different from SCPI.DISPlay.ECHO.DATA.

- Up to 10 data items can be displayed.
- Data is displayed as the declared data type without a cast.

Parameter	*V1,V2, ... ,V10*
Description	Data you want to display in the echo window.
Data type	Variant type (Variant)

Examples

```
Dim Nop As Long
Dim i As Integer
Dim Fdata As Variant
Nop = SCPI.SENSe(1).SWEep.POINts
Fdata = SCPI.CALCulate(1).SELected.DATA.FDATa
ECHO "Test Results"
For i=1 to Nop
    ECHO i, Fdata(2*i-2), Fdata(2*i-1)
Next i


ECHO SCPI.SYSTem.ERRor
```

Related objects

SCPI.DISPlay.ECHO.DATA

Equivalent key

No equivalent key is available on the front panel.

NAME

Object type

Property

Syntax

App = NAME

Description

Reads out the application name of VBA. E5061B is always read out. (Read only)

Variable

Parameter	*App*
Description	Application name
Data type	Character string type (String)

Examples

```
Dim Inst As String
Inst = NAME
ECHO Inst
```

Equivalent key

No equivalent key is available on the front panel.

Parse

Object type

Method (**Write-only**)

Syntax

Parse(*Scpi*)

Return = Parse(*Scpi?*)

Description

Executes an SCPI command of the E5061B.

The Parse object is a little slower in the execution speed than the COM object which has the same function as the SCPI command because it must parse the message string of the SCPI command.

Variable

Parameter	*Scpi*
Description	SCPI command
Data type	Character string type (String)

Parameter	*Return*
Description	Response (query) of the SCPI command
Data type	Character string type (String)

Examples

```
Dim Start As String
Parse(":SENS1:FREQ:STAR 100E6")
Start = Parse(":SENS1:FREQ:STAR?")
```

```
Dim TtlLbl As String
Parse(":DISP:WIND1:TITL:DATA ""filter""")
TtlLbl = Parse(":DISP:WIND1:TITL:DATA?")
```

```
Dim Fmt As String
Parse(":CALC1:PAR2:SEL")
```

```
Parse(":CALC1:FORM SMIT")
Fmt = Parse(":CALC1:FORM?")
```

```
Dim BckLght As String
Parse(":SYST:BACK OFF")
BckLght = Parse(":SYST:BACK?")
```

Equivalent key

No equivalent key is available on the front panel.

Prompt

Object type

Method (**Write-only**)

Syntax

Prompt(*Mes*)

Description

Displays the message you specify on the instrument status bar (at the bottom of the LCD display) and suspends the program until the **Macro Setup** > **Continue** button is pressed. (No read)

NOTE When using this object, execute the program with the Visual Basic application closed since you need to press the **Macro Setup** > **Continue**. For more information, see Running a Program from the E5061B Measurement Screen. If you need to abort the program, see Stopping with the Dialog Box Appeared.

Variable

Parameter	*Mes*
Description	Message
Data type	Character string type (String)

Examples

Prompt ("Connect DUT, and then press [Continue]")

Equivalent key

No equivalent key is available on the front panel.

UserMenu.Item(*Key_id*).Caption

Object type

Property

Syntax

UserMenu.Item(*Key_id*).Caption = *Lbl*

Lbl = UserMenu.Item(*Key_id*).Caption

Description

Sets the label name of the user menu function softkeys 1 to 10 (*Key_id*).

Variable

Parameter	*Key_id*
Description	Softkey number for the user menu function
Data type	Long integer type (Long)
Range	1 to 10
Note	You cannot omit this because it does not have a preset value. If the specified variable is out of the valid setting range, an error occurs when executed.

Parameter	*Lbl*
Description	Softkey label name for the user menu function
Data type	Character string type (String)
Preset value	Varies depending on the specified softkey number.

Examples

```
Dim KeyLbl As String
UserMenu.Item(1).Caption = "Meas"
KeyLbl = UserMenu.Item(1).Caption
```

Equivalent key

No equivalent key is available on the front panel.

UserMenu.Item(*Key_id*).Enabled

Object type

Property

Syntax

UserMenu.Item(*Key_id*).Enabled = *Status*

Status = UserMenu.Item(*Key_id*).Enabled

Description

Makes the user menu function softkeys 1 to 10 (*Key_id*) enabled/disabled. The softkey label disabled is displayed in grey color and its softkey cannot be pressed.

Variable

Parameter	*Status*
Description	Enabled/disabled for the user menu function softkey
Data type	Boolean type (Boolean)
Range	Select from the following. • True or -1 Makes the softkey enabled. • False or 0 Makes the softkey enabled.
Preset value	True or -1

For information on the variable (*Key_id*), see UserMenu.Item.Caption.

Examples

```
Dim KeyEna As Boolean
UserMenu.Item(10).Enabled = False
KeyEna = UserMenu.Item(10).Enabled
```

Related objects

UserMenu.Press

Equivalent key

No equivalent key is available on the front panel.

UserMenu.PRESet

Object type

Method (**Write-only**)

Syntax

UserMenu.PRESet

Description

Presets the label name and enables/disables settings for the user menu softkeys. (No read)

Examples

UserMenu.PRESet

Related objects

UserMenu.Item.Caption

UserMenu.Item.Enabled

Equivalent key

Macro Setup > **Preset User Menu**

UserMenu.Press(*Key_id*)

Object type

Method (**Write-only**)

Syntax

UserMenu.Press(*Key_id*)

Description

Presses one of the user menu function softkeys 1 to 10 (*id*). (No read)

Variable

For information on the variable (*Key_id*), see UserMenu.Item.Caption.

Examples

```
UserMenu.Press(1)
```

Related objects

UserMenu.Item.Enabled

Equivalent key

Macro Setup > **User Menu** > **Button 1** to **Button 10**

UserMenu.Show

Object type

Method (**Write-only**)

Syntax

UserMenu.Show

Description

Displays the user menu function softkeys in the softkey area. (No read)

Examples

UserMenu.Show

Equivalent key

Macro Setup > **User Menu**

UserMenu_OnPress(ByVal *Key_id* As Long)

Object type

Event

Description

Executes the processing when one of the user menu function softkeys 1 to 10 (*Key_id*) are pressed. Write the processing in the "UserMenu" object. For more information on its use, see Executing a Procedure with a Softkey (User Menu Function).

Variable

For information on the variable (*Key_id*), see UserMenu.Item.Caption.

Examples

```
Private Sub UserMenu_OnPress(ByVal id As Long)
If id = 1 Then
MsgBox "Button 1 was pressed."
ElseIf id = 10 Then
MsgBox "Button 10 was pressed."
End If
End Sub
```

Equivalent key

No equivalent key is available on the front panel.

VBAVersion

Object type

Property

Syntax

Vers = VBAVersion

Description

Reads out the version information of VBA installed in the E5061B. (Read only)

Variable

Parameter	*Vers*
Description	VBA version information
Data type	Character string type (String)

Examples

```
Dim Version As String
Version = VBAVersion
ECHO Version
```

Equivalent key

From the **Help** menu of the Visual Basic editor, click **About Microsoft Visual Basic...**.

WaitOnSRQ

Object type

Method (**Write-only**)

Syntax

WaitOnSRQ *Status*, *Timeout*

Description

Suspends the program for a specified time until the RQS/MSS bit (bit 6) of the status byte register changes to 1. (No read)

Variable

Parameter	*Status*
Description	State of the RQS/MSS bit (read only)
Data type	Boolean type (Boolean)
Range	One of the following is returned. • True or -1 1 has been received within the specified time. • False or 0 1 has not been received within the specified time due to time-out or abort.

Parameter	*Timeout*
Description	Time-out time
Data type	Long integer type (Long)
Range	0 to 2,147,483,647
Preset value	-1 (infinity)
Unit	ms (millisecond)

Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Stat As Boolean
SCPI.IEEE4882.CLS
SCPI.STATus.OPERation.PTRansition = 0
SCPI.STATus.OPERation.NTRansition = 16
SCPI.STATus.OPERation.ENABle = 16
SCPI.IEEE4882.SRE = 128
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.INITiate(1).CONTinuous = True
SCPI.TRIGger.SEQuence.IMMediate
WaitOnSRQ Stat, 10000
If Stat = True Then
MsgBox "Done"
End If
```

Equivalent key

No equivalent key is available on the front panel.

ABORT

SCPI.ABORt

Object type

Method (**Write-only**)

Syntax

SCPI.ABORt

Description

This command aborts the measurement and changes the trigger sequence for all channels to idle state.

The channels for which the continuous startup mode is set to *ON* (setting to start up the trigger system continuously), changes immediately from idle to start-up state. See Trigger System for details.

Examples

SCPI.ABORt

Related objects

SCPI.INITiate(Ch).IMMediate

SCPI.INITiate(Ch).CONTinuous

Equivalent key

Trigger > **Restart**

Equivalent SCPI command

Syntax

:ABORt

Example of use

```
10 OUTPUT 717;":ABORt"
```

CALCULATE

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.C1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.C1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.C1

Description

This command sets or gets C1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	C1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Farad (F)
Resolution	1E-21 (1z)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.A.C1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.A.C1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **A**

Equivalent Circuit > **C1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of C1.

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}:EPARameters:CIRCuit:A:C1 <numeric>
:CALCulate{[1]-4}:EPARameters:CIRCuit:A:C1?
```

Query Response

```
<numeric><newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:A:C1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:A:C1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.L1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.L1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.L1

Description

This command sets or gets L1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	L1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Henry (H)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate4.EPARameters.CIRCuit.A.L1 = Var
Var = SCPI.CALCulate4.EPARameters.CIRCuit.A.L1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.C1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **A**

Equivalent Circuit > **L1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of L1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:A:L1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:A:L1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:A:L1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:A:L1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.R1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.R1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.R1

Description

This command sets and gets R1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	R1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Ohm (Ω)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate4.EPARameters.CIRCuit.A.R1 = Var
Var = SCPI.CALCulate4.EPARameters.CIRCuit.A.R1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.C1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.A.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **A**

Equivalent Circuit > **R1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of R1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:A:R1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:A:R1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:A:R1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:A:R1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.C1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.C1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.C1

Description

This command sets or gets C1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	C1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Farad (F)
Resolution	1E-21 (1z)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.C.C1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.C.C1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **C**

Equivalent Circuit > **C1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of C1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:C:C1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:C:C1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:C:C1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:C:C1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.L1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.L1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.L1

Description

This command sets or gets L1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	L1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Henry (H)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.C.L1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.C.L1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.C1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **C**

Equivalent Circuit > **L1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of L1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:C:L1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:C:L1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:C:L1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:C:L1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.R1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.R1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.R1

Description

This command sets or gets R1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	R1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Ohm (Ω)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.C.R1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.C.R1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.C1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.C.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **C**

Equivalent Circuit > **R1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of R1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:C:R1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:C:R1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:C:R1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:C:R1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.C1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.C1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.C1

Description

This command sets or gets C1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	C1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Farad (F)
Resolution	1E-21 (1z)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.B.C1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.B.C1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **B**

Equivalent Circuit > **C1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of C1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:B:C1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:B:C1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:B:C1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:B:C1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.L1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.L1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.L1

Description

This command sets or gets L1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	L1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Henry (H)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.B.L1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.B.L1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.C1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **B**

Equivalent Circuit > **L1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of L1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:B:L1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:B:L1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:B:L1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:B:L1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.R1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.R1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.R1

Description

This command sets and gets R1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	R1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Ohm (Ω)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.B.R1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.B.R1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.C1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.B.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **B**

Equivalent Circuit > **R1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of R1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:B:R1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:B:R1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:B:R1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:B:R1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.C1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.C1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.C1

Description

This command sets or gets C1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	C1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Farad (F)
Resolution	1E-21 (1z)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.D.C1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.D.C1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **D**

Equivalent Circuit > **C1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of C1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:D:C1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:D:C1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALCulate1:EPAR:CIRC:D:C1 0"
20 OUTPUT 717;":CALCulate1:EPAR:CIRC:D:C1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.L1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.L1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.L1

Description

This command sets or gets L1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	L1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Henry (H)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.D.L1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.D.L1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.C1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **D**

Equivalent Circuit > **L1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of L1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:D:L1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:D:L1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:D:L1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:D:L1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.R1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.R1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.R1

Description

This command sets or gets R1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	R1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Ohm (Ω)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.D.R1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.D.R1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.C1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.D.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **D**

Equivalent Circuit > **R1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of R1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:D:R1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:D:R1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:D:R1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:D:R1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C0

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C0 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C0

Description

This command sets or gets C0 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	C0
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Farad (F)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.E.C0 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.E.C0
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.L1

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **E**

Equivalent Circuit > **C0**

Using either the keyboard or ENTRY keys on the front panel, enter the value of C0.

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}:EPARameters:CIRCuit:E:C0 <numeric>
:CALCulate{[1]-4}:EPARameters:CIRCuit:E:C0?
```

Query Response

```
<numeric><newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:E:C0 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:E:C0?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C1

Description

This command sets or gets C1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	C1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Farad (F)
Resolution	1E-21 (1z)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.E.C1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.E.C1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.L1

SCPI.CALCulate(Ch).EPARameters.CIRCuit.E.C0

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **E**

Equivalent Circuit > **C1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of C1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:E:C1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:E:C1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:E:C1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:E:C1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.L1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.L1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.L1

Description

This command sets or gets L1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	L1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Henry (H)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.E.L1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.E.L1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.R1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C1

SCPI.CALCulate(Ch).EPARameters.CIRCuit.E.C0

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **E**

Equivalent Circuit > **L1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of L1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:E:L1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:E:L1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:E:L1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:E:L1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.R1

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.R1 = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.R1

Description

This command sets and gets R1 value of equivalent circuit parameters.

Variable

Parameter	*Value*
Description	R1
Data Type	Double precision floating point type (Double)
Range	-1E ~ 1E
Preset Value	0
Unit	Ohm (Ω)
Resolution	1E-18 (1a)

Examples

```
Dim Var as Double
Var= 0
SCPI.CALCulate(1).EPARameters.CIRCuit.E.R1 = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.E.R1
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.C1

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.E.L1

SCPI.CALCulate(Ch).EPARameters.CIRCuit.E.C0

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **E**

Equivalent Circuit > **R1**

Using either the keyboard or ENTRY keys on the front panel, enter the value of R1.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit:E:R1 <numeric>

:CALCulate{[1]-4}:EPARameters:CIRCuit:E:R1?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC:E:R1 0"
20 OUTPUT 717;":CALC1:EPAR:CIRC:E:R1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.TYPE

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.TYPE = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.CIRCuit.TYPE

Description

This command sets or gets Select Eqv Ckt Type.

Variable

Parameter	*Value*
Description	A|B|C|D|E
Data Type	Character string type (String)
Range	A|B|C|D|E
Preset Value	A
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var= "A"
SCPI.CALCulate(1).EPARameters.CIRCuit.TYPE = Var
Var = SCPI.CALCulate(1).EPARameters.CIRCuit.TYPE
```

Equivalent Key

Analysis > **Equivalent Circuit** > **Select Circuit** > **A** or

Analysis > **Equivalent Circuit** > **Select Circuit** > **B** or

Analysis > **Equivalent Circuit** > **Select Circuit** > **C** or

Analysis > **Equivalent Circuit** > **Select Circuit** > **D** or

Analysis > **Equivalent Circuit** > **Select Circuit** > **E**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:CIRCuit[:TYPE] {A|B|C|D|E}

:CALCulate{[1]-4}:EPARameters:CIRCuit[:TYPE]?

Query Response

{A|B|C|D|E}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:CIRC A"
20 OUTPUT 717;":CALC1:EPAR:CIRC?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).EPARameters.DISPlay.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.DISPlay.STATe = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.DISPlay.STATe

Description

This command displays the equivalent circuit parameter values.

Variable

Parameter	*Status*
Description	ON|OFF of the display of the equivalent circuit parameter
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the display of the equivalent circuit parameter. • False or OFF: Turns OFF the display of the equivalent circuit parameter.
Preset Value	False or OFF
Unit	-
Resolution	-

Examples

```
Dim Var as Boolean
SCPI.CALCulate(1).EPARameters.DISPlay.STATe = True
Var = SCPI.CALCulate(1).EPARameters.DISPlay.STATe
```

Equivalent Key

Analysis > **Equivalent Circuit** > **Display**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:DISPlay[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}:EPARameters:DISPlay[:STATe]?

Query Response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALCulate1:EPAR:DISP ON"
20 OUTPUT 717;":CALCulate1:EPAR:DISP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.EXECute

Object Type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.EXECute

Description

This command executes the equivalent circuit analysis in the selected equivalent circuit model in the partial search range.

Examples

SCPI.CALCulate(1).EPARameters.EXECute

Equivalent Key

Analysis > **Equivalent Circuit** > **Calculate**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:EXECute

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:EXEC"
```

SCPI.CALCulate(*Ch*).EPARameters.SIMulate.AUTO

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.SIMulate.AUTO = *Value*

Value = SCPI.CALCulate(*Ch*).EPARameters.SIMulate.AUTO

Description

This command executes the equivalent circuit analysis simulation function automatically when parameters are changed. When this is set at ON, if one of the parameter is changed or set, the simulation calculation is done automatically. If its set at OFF, the simulation calculation is not executed but the simulated Memory Trace is still shown.

Variable

Parameter	*Status*
Description	ON|OFF of the equivalent circuit analysis simulation function
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the equivalent circuit analysis simulation function. • False or OFF: Turns OFF the equivalent circuit analysis simulation function.
Preset Value	False or OFF
Unit	-
Resolution	-

Examples

```
Dim Var as Boolean
Var= True
SCPI.CALCulate(1).EPARameters.SIMulate.AUTO = Var
Var = SCPI.CALCulate(1).EPARameters.SIMulate.AUTO
```

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.SIMulate.IMMediate

Equivalent Key

There is no equivalent key is available on the front panel. However, the similar key is:

Analysis > **Equivalent Circuit** > **Simulate** > **ON|OFF**

When this softkey is turned ON, its equivalent to:

SCPI.CALCulate[1-4].EPARameters.SIMulate.AUTO ON +
SCPI.CALCulate(*Ch*).EPARameters.SIMulate.IMMediate

When this softkey is turned OFF, its equivalent to:

SCPI.CALCulate[1-4].EPARameters.SIMulate.AUTO OFF

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:SIMulate:AUTO {ON|OFF|1|0}

:CALCulate{[1]-4}:EPARameters:SIMulate:AUTO?

Query Response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:SIM:AUTO ON"
20 OUTPUT 717;":CALC1:EPAR:SIM:AUTO?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).EPARameters.SIMulate.IMMediate

Object Type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).EPARameters.SIMulate.IMMediate

Description

This command writes the simulation results into the memory trace, then set "Data & Mem" for the display setting. All traces related with impedance measurement are updated, even if CALCulate [1-4]: EPARameters: SIMulate: AUTO is set at OFF.

Examples

SCPI.CALCulate(1).EPARameters.SIMulate.IMMediate

Related Objects

SCPI.CALCulate(*Ch*).EPARameters.SIMulate.AUTO

Equivalent Key

There is no equivalent key is available on the front panel. However, the similar key is:

Analysis > **Equivalent Circuit** > **Simulate** > **ON|OFF**

When this softkey is turned ON, its equivalent to:

SCPI.CALCulate[1-4].EPARameters.SIMulate.AUTO ON +
SCPI.CALCulate(*Ch*).EPARameters.SIMulate.IMMediate

When this softkey is turned OFF, its equivalent to:

SCPI.CALCulate[1-4].EPARameters.SIMulate.AUTO OFF

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:EPARameters:SIMulate[:IMMediate]

Example of use

```
10 OUTPUT 717;":CALC1:EPAR:SIM"
```

SCPI.CALCulate(*Ch*).PARameter(*Tr*).DEFine

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).PARameter(*Tr*).DEFine = *Param*

Param = SCPI.CALCulate(*Ch*).PARameter(*Tr*).DEFine

Description

This command sets/gets the measurement parameter of the selected trace (*Tr*), for the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Measurement parameter
Data Type	Character string type (String)
Range	Select either one of the following: •S-Parameter: o S11|S21|S12|S22 o A|B|R1|R2 o Gain-Phase o TR|T|R o Z (Impedance Measurement)
Preset Value	"S11"

Examples

```
Dim MeasPara As String
SCPI.CALCulate(1).PARameter(1).DEFine = "s21"
MeasPara = SCPI.CALCulate(1).PARameter(1).DEFine
```

Related Objects

SCPI.CALCulate(Ch).SELected.ZPARameter.DEFine

SCPI.CALCulate(Ch).SELected.FORMat

Equivalent key

Meas > **S<XY>** {X=1-2;Y=1-2}

Meas > **Absolute** > **R1**|**R2**|**A**|**B**

Meas > **Gain-Phase** > **T/R**|**R**|**T**

Meas > **Impedance Analysis Menu**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}:PARameter{[1]-4}:DEFine
{S11|S21|S12|S22|A|B|R1|R2|TR|T|R}

:CALCulate{[1]-4}:PARameter{[1]-4}:DEFine?

Query response

{S11|S21|S12|S22|A|B|R1|R2|TR|T|R}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:PAR1:DEF S21"
20 OUTPUT 717;":CALC1:PAR1:DEF?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SELect

Object type

Method (**Write only**)

Syntax

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SELect

Description

This command sets the selected trace (*Tr*) of selected channel (*Ch*) to the active trace.

You can set a trace to be displayed only to the active trace. If this object is used to set a trace displayed to the non-active trace, an error occurs during execution and the object is ignored. (No read)

Variable

None

Examples

SCPI.CALCulate(2).PARameter(2).SELect

Related objects

SCPI.CALCulate(Ch).SELected.CORRection.EDELay.TIME

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

Equivalent key

Trace Prev

Trace Next

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}:PARameter{[1]-4}:SELect

Example of use

```
10 OUTPUT 717;":CALC2:PAR2:SEL"
```

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SPORt

Type of object

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).PARameter(*Tr*).SPORt = *Value*

Value = SCPI.CALCulate(*Ch*).PARameter(*Tr*).SPORt

Description

This command sets/gets the output port used for absolute for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Setting of the output port
Data type	Long integer type (Long)
Range	1 to 2
Preset value	1
Note	You need to set the measurement parameter for absolute measurements with the SCPI.CALCulate(Ch).PARameter(Tr).DEFine command.

Example of use

```
Dim Sport As Long
SCPI.CALCulate(1).PARameter(1).DEFine = "B"
SCPI.CALCulate(1).PARameter(1).SPORt = 2
Sport = SCPI.CALCulate(1).PARameter(1).SPORt
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).DEFine

Equivalent key

Meas > **Absolute** > **A(x)** , **B(x)**, **R1(x)**, **R2(x)** (x: 1 to 2)

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}:PARameter{[1]-4}:SPORt <numeric>

:CALCulate{[1]-4}:PARameter{[1]-4}:SPORt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:PAR1:DEF B"
20 OUTPUT 717;":CALC1:PAR1:SPOR 2"
30 OUTPUT 717;":CALC1:PAR1:SPOR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).PARameter.COUNt

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).PARameter.COUNt = *Value*

Value = SCPI.CALCulate(*Ch*).PARameter.COUNt

Description

This command sets/gets the number of traces of selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of traces
Data type	Long integer type (Long)
Range	1 to 8, Varies depending on the upper limit setting for the channel/trace number
Preset value	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim TraceNum As Long
SCPI.CALCulate(1).PARameter.COUNt = 4
TraceNum = SCPI.CALCulate(1).PARameter.COUNt
```

Equivalent key

Display > **Num of Traces**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}:PARameter:COUNt <numeric>

:CALCulate{[1]-4}:PARameter:COUNt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:PAR:COUN 4"
20 OUTPUT 717;":CALC1:PAR:COUN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.DB

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.BLIMit.DB = Value

Value = SCPI.CALCulate(*Ch*).SELected.BLIMit.DB

Description

This command sets/gets the bandwidth threshold value (attenuation from the peak) of the bandwidth test, for the selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Value*
Description	Bandwidth N dB points.
Data type	Double precision floating point type (Double)
Range	0 to 5E8
Preset value	3
Unit	Varies depending on the selected data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): ° (degree) • Group delay (GDEL): s (second) • Others: No unit

Examples

```
Dim BLimDB As Double
SCPI.CALCulate(1).SELected.BLIMit.DB = 3
BLimDB = SCPI.CALCulate(1).SELected.BLIMit.DB
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

Equivalent key

Analysis > Bandwidth Limit > N dB Points

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:BLIMit:DB <numeric>

:CALCulate{[1]-4}[:SELected]:BLIMit:DB?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:BLIM:DB 3"
20 OUTPUT 717;":CALC1:BLIM:DB?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.DISPlay.MARKer

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.BLIMit.DISPlay.MARKer = Status

Status = SCPI.CALCulate(*Ch*).SELected.BLIMit.DISPlay.MARKer

Description

This command turns ON/OFF the marker display of the bandwidth test, for the active trace of selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Status*
Description	ON/OFF of the bandwidth marker.
Data type	Boolean type (Boolean)
Range	Select from the following. • True or ON: Turns ON the bandwidth marker. • False or OFF: Turns OFF the bandwidth marker.
Preset value	False or OFF

Examples

```
Dim BLimMk As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.BLIMit.DISPlay.MARKer = True
BLimMk = SCPI.CALCulate(1).SELected.BLIMit.DISPlay.MARKer
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.VALue

Equivalent key

Analysis > **Bandwidth Limit** > **BW Marker**

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:BLIMit:DISPlay:MARKer {ON|OFF|1|0}
:CALCulate{[1]-4}[:SELected]:BLIMit:DISPlay:MARKer?
```

Query response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:BLIM:DISP:MARK ON"
20 OUTPUT 717;":CALC1:BLIM:DISP:MARK?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.DISPlay.VALue

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.BLIMit.DISPlay.VALue = Status

Status = SCPI.CALCulate(*Ch*).SELected.BLIMit.DISPlay.VALue

Description

This command turns ON/OFF the bandwidth value display of the bandwidth test, for the active trace of selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Status*
Description	ON/OFF of the bandwidth display of the bandwidth test.
Data type	Boolean type (Boolean)
Range	Select from the following. • True or ON: Turns ON the bandwidth display. • False or OFF: Turns OFF the bandwidth display.
Preset value	False or OFF

Examples

```
Dim BLimVal As Boolean
SCPI.CALCulate(1).PARameter(1).SELectSCPI.CALCulate(1).SELected.BLIMit.DISPlay.VALue = True
BLimVal = SCPI.CALCulate(1).SELected.BLIMit.DISPlay.VALue
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer

Equivalent key

Analysis > **Bandwidth Limit** > **BW Display**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:BLIMit:DISPlay:VALue {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:BLIMit:DISPlay:VALue?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:BLIM:DISP:VAL ON"
20 OUTPUT 717;":CALC1:BLIM:DISP:VAL?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.FAIL

Object type

Property (**Read Only**)

Syntax

Status = SCPI.CALCulate(*Ch*).SELected.BLIMit.FAIL

Description

This command gets the bandwidth limit test results, for the active trace of the selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Status*
Description	The bandwidth limit test result
Data type	Long integer type (Long)
Range	Returns from the following. • 0: The bandwidth limit test result is PASS. • 1: The bandwidth limit test result is FAIL of wide. • -1: The bandwidth limit test result is FAIL of narrow.
Note	When the bandwidth limit test is set to OFF, the 0 is always read out.

Examples

```
Dim Result As Integer
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.BLIMit.STATe = 1
Result = SCPI.CALCulate(1).SELected.BLIMit.FAIL
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:BLIMit:FAIL?

Query response

{0|1|-1}<newline><^END>

(0: Pass, 1: Fail (Wide), -1:Fail (Narrow), When the bandwidth limit test is set to OFF, 0 is always read out.)

Example of use

```
10 OUTPUT 717;":CALC1:BLIM:FAIL?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.MAXimum

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.BLIMit.MAXimum = Value

Value = SCPI.CALCulate(*Ch*).SELected.BLIMit.MAXimum

Description

This command sets/gets the upper limit value of the bandwidth test, for the selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Value*
Description	Maximum bandwidth
Data type	Double precision floating point type (Double)
Range	0 to 1E12
Preset value	3E5
Unit	Hz (hertz), dBm or second

Examples

```
Dim BLimMax As Double
SCPI.CALCulate(1).SELected.BLIMit.MAXimum = 1E9
BLimMax = SCPI.CALCulate(1).SELected.BLIMit.MAXimum
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

SCPI.CALCulate(Ch).SELected.BLIMit.MINimum

Equivalent key

Analysis > **Bandwidth Limit** > **Max Bandwidth**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:BLIMit:MAXimum <numeric>

:CALCulate{[1]-4}[:SELected]:BLIMit:MAXimum?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:BLIM:MAX 3E5"
20 OUTPUT 717;":CALC1:BLIM:MAX?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.MINimum

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.BLIMit.MINimum = Value

Value = SCPI.CALCulate(*Ch*).SELected.BLIMit.MINimum

Description

This command sets/gets the lower limit value of the bandwidth test, for the selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Value*
Description	Minimum bandwidth
Data type	Double precision floating point type (Double)
Range	0 to 1E12
Preset value	1E4
Unit	Hz (hertz), dBm or second

Examples

```
Dim BLimMin As Double
SCPI.CALCulate(1).SELected.BLIMit.MINimum = 1E6
BLimMin = SCPI.CALCulate(1).SELected.BLIMit.MINimum
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum

Equivalent key

Analysis > **Bandwidth Limit** > **Min Bandwidth**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:BLIMit:MINimum <numeric>

:CALCulate{[1]-4}[:SELected]:BLIMit:MINimum?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:BLIM:MIN 1E4"
20 OUTPUT 717;":CALC1:BLIM:MIN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.REPort.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.BLIMit.REPort.DATA

Description

This command reads the bandwidth value of the bandwidth test, for the active trace of selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Data*
Description	The bandwidth value of the bandwidth
Data type	Double precision floating point type (Double)

Examples

```
Dim BWData As Double
SCPI.CALCulate(1).PARameter(1).SELect
BWData = SCPI.CALCulate(1).SELected.BLIMit.REPort.DATA
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:BLIMit:REPort[:DATA]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:BLIM:REP?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.BLIMit.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.BLIMit.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.BLIMit.STATe

Description

This command turns ON/OFF the bandwidth test function, for the active trace of selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Status*
Description	ON/OFF the bandwidth test function.
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the bandwidth test function. • False or OFF: Turns OFF the bandwidth test function.
Preset value	False or OFF

Examples

```
Dim BLimTest As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.BLIMit.STATe = True
BLimTest = SCPI.CALCulate(1).SELected.BLIMit.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.BLIMit.DB

SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.MARKer

SCPI.CALCulate(Ch).SELected.BLIMit.DISPlay.VALue

SCPI.CALCulate(Ch).SELected.BLIMit.FAIL

SCPI.CALCulate(Ch).SELected.BLIMit.MAXimum

SCPI.CALCulate(Ch).SELected.BLIMit.MINimum

SCPI.CALCulate(Ch).SELected.BLIMit.REPort.DATA

Equivalent key

Analysis > **Bandwidth Limit** > **BW Test**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:BLIMit[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:BLIMit[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:BLIM ON"
20 OUTPUT 717;":CALC1:BLIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.CONVersion.FUNCtion

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.CONVersion.FUNCtion = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.CONVersion.FUNCtion

Description

This command sets/gets the parameter after conversion using the parameter conversion function, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The parameter after conversion
Data type	Character string type (String)
Range	Select from the following: • "ZREFlection": Specifies the equivalent impedance in reflection measurement. • "ZTRansmit": Specifies the equivalent impedance (series) in transmission measurement. • "YREFlection": Specifies the equivalent admittance in reflection measurement. • "YTRansmit": Specifies the equivalent admittance (series) in transmission measurement. • "INVersion": Specifies the inverse S-parameter. • "ZTSHunt": Specifies the equivalent impedance (shunt) in transmission measurement. • "YTSHunt": Specifies the equivalent admittance (shunt) in transmission measurement. • "CONJugation": Specifies the conjugate.
Preset value	"ZREFlection"

Examples

```
Dim Func As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.CONVersion.FUNCtion = "ztr"
Func = SCPI.CALCulate(1).SELected.CONVersion.FUNCtion
```

Related objects

SCPI.CALCulate(Ch).SELected.CONVersion.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

Equivalent key

Analysis > **Conversion** > **Function** > **Z:Reflection**|**Z:Transmission**|**Y:Reflection**|**Y:Transmission**|**1/S**| **Z:Trans-Shunt**|**Y:Trans-Shunt**|**Conjugation**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:CONVersion:FUNCtion {ZREFlection|ZTRansmit|YREFlection|YTRansmit|INVersion|ZTSHunt|YTSHunt|CONJugation}

:CALCulate{[1]-4}[:SELected]:CONVersion:FUNCtion?

Query response

{ZREF|ZTR|YREF|YTR|INV|ZTSH|YTSH|CONJ}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:CONV:FUNC ZTR"
20 OUTPUT 717;":CALC1:CONV:FUNC?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.CONVersion.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.CONVersion.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.CONVersion.STATe

Description

This command turns ON/OFF the parameter conversion function, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the parameter conversion function
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the parameter conversion function. • False or OFF: Turns OFF the parameter conversion function.
Preset value	False or OFF

Examples

```
Dim Conv As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.CONVersion.STATe = True
Conv = SCPI.CALCulate(1).SELected.CONVersion.STATe
```

Related objects

SCPI.CALCulate(Ch).SELected.CONVersion.FUNCtion

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > **Conversion** > **Conversion**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:CONVersion[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:CONVersion[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:CONV ON"
20 OUTPUT 717;":CALC1:CONV?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.TIME

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.TIME = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.CORRection.EDELay.TIME

Description

This command sets/gets the electrical delay time of the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Electrical delay time
Data type	Double precision floating point type (Double)
Range	-10 to 10
Preset value	0
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Edel As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.CORRection.EDELay.TIME = 0.2
Edel = SCPI.CALCulate(1).SELected.CORRection.EDELay.TIME
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Scale > **Electrical Delay** > **Electrical Delay**

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:CORRection:EDELay:TIME <numeric>
:CALCulate{[1]-4}[:SELected]:CORRection:EDELay:TIME?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:CORR:EDEL:TIME 0.2"
20 OUTPUT 717;":CALC1:CORR:EDEL:TIME?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.CORRection.OFFSet.PHASe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.CORRection.OFFSet.PHASe = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.CORRection.OFFSet.PHASe

Description

This command sets/gets the phase offset of the active trace of selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Phase offset
Data type	Double precision floating point type (Double)
Range	-360 to 360
Preset value	0
Unit	° (degree)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Offset As Double
SCPI.CALCulate(2).PARameter(1).SELect
SCPI.CALCulate(2).SELected.CORRection.OFFSet.PHASe = 2.5
Offset = SCPI.CALCulate(2).SELected.CORRection.OFFSet.PHASe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Scale > **Phase Offset**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:CORRection:OFFSet:PHASe <numeric>
:CALCulate{[1]-4}[:SELected]:CORRection:OFFSet:PHASe?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:CORR:OFFS:PHAS 2.5"
20 OUTPUT 717;":CALC1:CORR:OFFS:PHAS?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.DATA.FDATa

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.DATA.FDATa = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.FDATa

Description

This command sets/gets the formatted data array, for the active trace of selected channel (*Ch*).
The array data element varies in the data format (specified with the SCPI.CALCulate(Ch).SELected.FORMat object). For more information on the formatted data array, see Internal Data Processing.

NOTE If valid data is not calculated because of the invalid measurement, "1.#QNB" is read out.

Variable

Parameter	*Data*
Description	Indicates the array data (formatted data array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. • *Data(n×2-2) :*Data (primary value) at the n-th measurement point. • *Data(n×2-1) :*Data (secondary value) at the n-th measurement point. Always 0 when the data format is not the Smith chart format or the polar format. The index of the array starts from 0.
Data type	Variant type (Variant)
Note	If there is no array data of NOP (number of measurement point))×2 when setting a formatted data array, an error occurs when executed and the object is ignored.

Examples

```
Dim FmtData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
SCPI.CALCulate(1).PARameter(1).SELect
```

```
FmtData = SCPI.CALCulate(1).SELected.DATA.FDATa
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.DATA.FDATa = FmtData
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.SENSe(Ch).SWEep.POINts

SCPI.CALCulate(Ch).SELected.FORMat

SCPI.CALCulate(Ch).SELected.DATA.FMEMory

SCPI.CALCulate(Ch).SELected.DATA.SDATa

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:DATA:FDATa <numeric1>,… ,<numeric NOP×2>

:CALCulate{[1]-4}[:SELected]:DATA:FDATa?

Query response

{numeric 1},… ,{numeric NOP×2}<newline><^END>

Example of use

```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:FDAT?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.DATA.FMEMory

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.DATA.FMEMory = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.FMEMory

Description

This command sets/gets the formatted memory array, for the active trace of selected channel (*Ch*).

The array data element varies in the data format (specified with the SCPI.CALCulate(Ch).SELected.FORMat object). For more information on the formatted memory array, see Internal Data Processing.

NOTE If valid data is not calculated because of the invalid measurement, "1.#QNB" is read out.

Variable

Parameter	*Data*
Description	Indicates the array data (formatted memory array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. • *Data(n×2-2) :*Data (primary value) at the n-th measurement point. • *Data(n×2-1) :*Data (secondary value) at the n-th measurement point. Always 0 when the data format is not the Smith chart format or the polar format. The index of the array starts from 0.
Data type	Variant type (Variant)
Note	If there is no array data of NOP (number of measurement point))*2 when setting a formatted memory array, an error occurs when executed and the object is ignored.

Examples

```
Dim FmtMem As Variant
SCPI.SENSe(1).SWEep.POINts = 201
```

```
SCPI.CALCulate(1).PARameter(1).SELect
FmtMem = SCPI.CALCulate(1).SELected.DATA.FMEMory
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.DATA.FMEMory = FmtMem
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.SENSe(Ch).SWEep.POINts

SCPI.CALCulate(Ch).SELected.FORMat

SCPI.CALCulate(Ch).SELected.DATA.FDATa

SCPI.CALCulate(Ch).SELected.DATA.SMEMory

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:DATA:FMEMory <numeric 1>,… ,<numeric NOP×2>

:CALCulate{[1]-4}[:SELected]:DATA:FMEMory?

Query response

{numeric 1},… ,{numeric NOP×2}<newline><^END>

Example of use

```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:FMEM?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.DATA.SDATa

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.DATA.SDATa = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.SDATa

Description

This command sets/gets the corrected data array, for the active trace of selected channel (*Ch*).

For more information on the corrected data array, see Internal Data Processing

NOTE If valid data is not calculated because of the invalid measurement, "1.#QNB" is read out.

Variable

Parameter	*Data*
Description	Indicates the array data (corrected data array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. • *Data(n×2-2) :*Real part of the data (complex number) at the n-th measurement point. • *Data(n×2-1)* :Imaginary part of the data (complex number) at the n-th measurement point. The index of the array starts from 0.
Data type	Variant type (Variant)
Note	If there is no array data of NOP (number of measurement point))× 2 when setting a corrected data array, an error occurs when executed and the object is ignored.

Examples

```
Dim CorData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
CorData = SCPI.CALCulate(1).SELected.DATA.SDATa
SCPI.SENSe(2).SWEep.POINts = 201
SCPI.CALCulate(2).SELected.DATA.SDATa = CorData
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.SENSe(Ch).SWEep.POINts

SCPI.CALCulate(Ch).SELected.DATA.SMEMory

SCPI.CALCulate(Ch).SELected.DATA.FDATa

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:DATA:SDATa <numeric 1>,… ,<numeric NOP×2>

:CALCulate{[1]-4}[:SELected]:DATA:SDATa?

Query response

<numeric 1>,… ,<numeric NOP×2><^END>

Example of use

```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:SDAT?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.DATA.SMEMory

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.DATA.SMEMory = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.DATA.SMEMory

Description

This command sets/gets the corrected memory array, for the active trace of selected channel (*Ch*).

For more information on the corrected memory array, see Section Internal Data Processing.

NOTE If valid data is not calculated because of the invalid measurement, "1.#QNB" is read out.

Variable

Parameter	*Data*
Description	Indicates the array data (corrected memory array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. • *Data(n×2-2)* :Real part of the data (complex number) at the n-th measurement point. • *Data(n×2-1)* :Imaginary part of the data (complex number) at the n-th measurement point. The index of the array starts from 0.
Data type	Variant type (Variant)
Note	If there is no array data of NOP (number of measurement point))×2 when setting a corrected memory array, an error occurs when executed and the object is ignored.

Examples

```
Dim CorMem As Variant
SCPI.SENSe(1).SWEep.POINts = 201
CorMem = SCPI.CALCulate(1).SELected.DATA.SMEMory
SCPI.SENSe(2).SWEep.POINts = 201
SCPI.CALCulate(1).SELected.DATA.SMEMory = CorMem
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.SENSe(Ch).SWEep.POINts

SCPI.CALCulate(Ch).SELected.DATA.SDATa

SCPI.CALCulate(Ch).SELected.DATA.FMEMory

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:DATA:SMEMory <numeric 1>,… ,<numeric NOP×2>

:CALCulate{[1]-4}[:SELected]:DATA:SMEMory?

Query response

<numeric 1>,… ,<numeric NOP×2><^END>

Example of use

```
10 DIM A(1:201,1:2)
20 OUTPUT 717;":CALC1:DATA:SMEM?"
30 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.DATA.XAXis

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.DATA.XAXis

Description

This command reads the data of measurement points of X axis, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates the array data (measurement points) of X axis
Data type	Variant type (Variant)

Examples

```
Dim AnaData As Variant
AnaData = SCPI.CALC1.SEL.DATA.XAX
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:DATA:XAXis?

Query response

<numeric 1>,... ,<numeric N><^END>

Example of use

```
10 OUTPUT 717;":CALC1:DATA:XAX?"
20 ENTER 717;A(*)
```

SCPI.CALCulate(*Ch*).SELected.EQUation.STATE

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.EQUation.STATE= *Status*

Status = SCPI.CALCulate(*Ch*).SELected.EQUation.STATE

Description

This command enables/disables the Equation Editor of selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Sets/Gets the state of equation in the Equation Editor
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the Equation Editor. • False or OFF: Turns OFF the Equation Editor.
Preset value	False or OFF

Examples

```
Dim strEq As String
SCPI.CALCulate(1).SELected.EQUation.STATE = True
EqState = SCPI.CALCulate(1).SELected.EQUation.State
SCPI.CALCulate(1).SELected.EQUation.TEXT = "Example=S21/(1-S11)"
strEq = SCPI.CALCulate(1).SELected.EQUation.TEXT
```

Related objects

SCPI.CALCulate(Ch).SELected.EQUation.TEXT

SCPI.CALCulate(Ch).SELected.EQUation.VALid

Equivalent key

Display > **Equation**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:EQUation:STATE {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:EQUation:STATE?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EQU:STATE ON"
20 OUTPUT 717;":CALC1:EQU:STATE?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.EQUation.TEXT

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.EQUation.TEXT = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.EQUation.TEXT

Description

This command sets/gets the equation in the Equation Editor for selected channel (*Ch*). For valid parameters that can be used in this equation, refer to the Equation Editor.

Variable

Parameter	*Data*
Description	Sets/Gets the equation in the Equation Editor
Data type	Character string type (String)

Examples

```
Dim  strEq As String
SCPI.CALCulate(1).SELected.EQUation.STATE = True
EqState = SCPI.CALCulate(1).SELected.EQUation.State
SCPI.CALCulate(1).SELected.EQUation.TEXT = "Example=S21/(1-S11)"
strEq = SCPI.CALCulate(1).SELected.EQUation.TEXT
```

Related objects

SCPI.CALCulate(Ch).SELected.EQUation.STATE

SCPI.CALCulate(Ch).SELected.EQUation.VALid

Equivalent key

No equivalent key is available on the front panel for Equation Text but Equation Editor can be accessed through **Display** > **Equation Editor**.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:EQUation:TEXT <string1>

:CALCulate{[1]-4}[:SELected]:EQUation:TEXT?

Query response

<string 1><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EQU:TEXT?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.EQUation.VALid

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.EQUation.VALid

Description

This command returns False when the equation expression and label are correct but the required S-parameter data is not measured or if it refers the invalid corrected memory array. Annotation of '**Equ!**' is displayed when this command returns a False value.

NOTE Equation Editor can refer S parameter data and data present in corrected memory array.

Variable

Parameter	*Data*
Description	A boolean value which gets the state of the equation in the Equation Editor of selected channel (*Ch*) as invalid (False) or valid (True)
Data type	Boolean type (Boolean)
Range	True or ON: Valid Spara data False or OFF: Invalid Spara data
Preset value	False or OFF

Examples

```
Dim EqState As Boolean
EqState = SCPI.CALCulate(1).SELected.EQUation.VALid
```

Related objects

SCPI.CALCulate(Ch).SELected.EQUation.STATE

SCPI.CALCulate(Ch).SELected.EQUation.TEXT

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:EQUation:VALid?

Query response

<1|0><newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:EQU:VAL?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.CENTer

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.CENTer = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.CENTer

Description

This command sets/gets the center value of the gate used for the gating function of the fault location display, for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The center value of the gate
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	0 m or 0 ft
Unit	ft (feet) or m (meters)
Resolution	-

Examples

```
Dim Var as Double
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.CENTer = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.CENTer
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SHAPe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STARt

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STATe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.TYPE

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.UNIT

Equivalent Key

Analysis > Gating > Center

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:CENTer <numeric>

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:CENTer?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:DIST:CENT 0"
20 OUTPUT 717;":CALC1:FILT:GATE:DIST:CENT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.SHAPe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.SHAPe = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.SHAPe

Description

This command sets/gets the shape of the gate used for the gating function of the fault location display, for the active trace of the selected channel (*Ch)*.

Variable

Parameter	*Param*
Description	The shape of the gate
Data Type	Character string type (String)
Range	Select from the following: • "MAXimum": Specifies the maximum shape. • "WIDE": Specifies the wide shape. • "NORMal": Specifies the normal shape. • "MINimum": Specifies the minimum shape.
Preset Value	NORMal

Examples

```
Dim Var as String
Var= "MAXimum"
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.SHAPe = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.SHAPe
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STARt

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STATe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.TYPE

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.UNIT

Equivalent Key

Analysis > **Gating** > **Shape** > **Maximum**|**Wide**|**Normal**|**Minimum**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:SHAPe {MAXimum|WIDE|NORMal|MINimum}

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:SHAPe?

Query Response

{MAX|WIDE|NORM|MIN} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:DIST:SHAP WIDE"
20 OUTPUT 717;":CALC1:FILT:GATE:DIST:SHAP?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.SPAN

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.SPAN = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.SPAN

Description

This command sets/gets the span value of the gate used for the gating function of the fault location display, for the active trace of the selected channel (*Ch)*.

Variable

Parameter	*Value*
Description	the span value of the gate
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	19.6714211286 ft or 5.99584916 m
Unit	ft (feet) or m (meters)

Examples

```
Dim Var as Double
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.SPAN = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.SPAN
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SHAPe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STARt

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STATe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.TYPE

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.UNIT

Equivalent Key

Analysis > Gating > Span

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:SPAN <numeric>

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:SPAN?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALCulate1:FILT:GATE:DIST:SPAN 0"
20 OUTPUT 717;":CALCulate1:FILT:GATE:DIST:SPAN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STARt

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STARt

Description

This command sets/gets the start value of the gate used for the gating function of the fault location display, for the active trace of the selected channel (*Ch)*.

Variable

Parameter	*Value*
Description	The start value of the gate
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	-9.8357105643 ft or -2.99792458 m
Unit	ft (feet) or m (meters)

Examples

```
Dim Var as Double
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.STARt = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.STARt
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SHAPe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STATe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.TYPE

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.UNIT

Equivalent Key

Analysis > Gating > Start

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:STARt <numeric>
:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:STARt?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:DIST:STAR 0"
20 OUTPUT 717;":CALC1:FILT:GATE:DIST:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STATe

Description

This command turns ON/OFF the gating function of the fault location display, for the active trace of the selected channel (*Ch)*.

Variable

Parameter	*Status*
Description	ON/OFF of the gating function
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the gating function. • False or OFF: Turns OFF the gating function.
Preset Value	False or OFF

Examples

```
Dim Var as Boolean
Var= True
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.STATe = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.STATe
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SHAPe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STARt

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.TYPE

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.UNIT

Equivalent Key

Analysis > **Gating** > **Gating**

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:STATe
{ON|OFF|1|0}
:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:STATe?
```

Query Response

```
{1|0} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:DIST:STAT ON"
20 OUTPUT 717;":CALC1:FILT:GATE:DIST:STAT?"
30 ENTER 717;A
```

(SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STOP

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.STOP

Description

This command sets/gets the stop value of the gate used for the gating function of the fault location display, for the active trace of the selected channel (*Ch).*

Variable

Parameter	*Value*
Description	The stop value of the gate
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	9.8357105643 ft or 2.99792458 m
Unit	ft (feet) or m (meters)

Examples

```
Dim Var as Double
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.STOP = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.STOP
```

Related Objects

Equivalent Key

Analysis > **Gating** > **Stop**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:STOP <numeric>

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:STOP?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:DIST:STOP 0"
20 OUTPUT 717;":CALC1:FILT:GATE:DIST:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.TYPE

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.TYPE

Description

This command sets/gets the gate type used for the gating function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The gate type used for the gating function
Data Type	Character string type (String)
Range	Select from the following. • "BPASs": Specifies the band-pass type. • "NOTCh": Specifies the notch type.
Preset Value	"BPASs"

Examples

```
Dim Var as String
Var= "BPASs"
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.TYPE = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.TYPE
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SHAPe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STARt

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STATe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.UNIT

Equivalent Key

Analysis > **Gating** > **Type**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:TYPE {BPASs|NOTCh}

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:TYPE?

Query Response

{BPAS|NOTC} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:DIST:TYPE NOTC"
20 OUTPUT 717;":CALC1:FILT:GATE:DIST:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.UNIT

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.UNIT = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.DISTance.UNIT

Description

This command sets/gets the distance unit used for the gating function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The distance unit used for the gating function
Data Type	Character string type (String)
Range	Select from the following. • "METers": Specifies the meter. • "FEET": Specifies the feet.
Preset Value	"METers"

Examples

```
Dim Var as String
Var= "METers"
SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.UNIT = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.DISTance.UNIT
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SHAPe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STARt

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STATe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.TYPE

Equivalent Key

Analysis > **Gating** > **Unit** > **Meters**|**Feet**|**Seconds**

NOTE When performing this operation from the front panel, you select the horizontal axis of the gating function as either time or distance at the same time.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:UNIT {METers|FEET}

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:DISTance:UNIT?

Query Response

{MET|FEET} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:DIST:UNIT MET"
20 OUTPUT 717;":CALC1:FILT:GATE:DIST:UNIT?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.METHod

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.METHod = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.METHod

Description

This command sets/gets the horizontal axis of the gating function either time or distance, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The horizontal axis of the gating function
Data Type	Character string type (String)
Range	Select from the following. • "TIME": Specifies the time. • "DISTance": Specifies the distance.
Preset Value	"TIME"

Examples

```
Dim Var as String
Var= "TIME"
SCPI.CALCulate(4).SELected.FILTer.GATE.METHod = Var
Var = SCPI.CALCulate(4).SELected.FILTer.GATE.METHod
```

Related Objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.DISTance.UNIT

Equivalent Key

Analysis > **Gating** > **Unit** > **Meters**|**Feet**|**Seconds**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:METHod {TIME|DISTance}

:CALCulate{[1]-4}[:SELected]:FILTer:GATE:METHod?

Query Response

{TIME|DIST} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:GATE:METH TIME"
20 OUTPUT 717;":CALC1:FILT:GATE:METH?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.CENTer

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.CENTer = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.CENTer

Description

This command sets/gets the center value of the gate used for the gating function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The center value of the gate
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	0
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim FilCent As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.CENTer = 1E-8
FilCent = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.CENTer
```

Related objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SPAN

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Gating > Center

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:CENTer <numeric>

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:CENTer?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:TIME:CENT 1E-8"
20 OUTPUT 717;":CALC1:FILT:TIME:CENT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SHAPe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SHAPe = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SHAPe

Description

This command sets/gets the shape of the gate used for the gating function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The shape of the gate
Data type	Character string type (String)
Range	Select from the following: • "MAXimum": Specifies the maximum shape. • "WIDE": Specifies the wide shape. • "NORMal": Specifies the normal shape. • "MINimum": Specifies the minimum shape.
Preset value	"NORMal"

Examples

```
Dim FilShape As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe = "wide"
FilShape = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe
```

Related objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.TYPE

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Gating > Shape > Maximum|Wide|Normal|Minimum

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:SHAPe {MAXimum|WIDE|NORMal|MINimum}

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:SHAPe?

Query response

{MAX|WIDE|NORM|MIN}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:TIME:SHAP WIDE"
20 OUTPUT 717;":CALC1:FILT:TIME:SHAP?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SPAN

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SPAN = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.SPAN

Description

This command sets/gets the span value of the gate used for the gating function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The span value of the gate
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	2E-8
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim FilStar As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SPAN = 1E-8
FilStar = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SPAN
```

Related objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.CENTer

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Gating > Span

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:SPAN <numeric>

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:SPAN?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:TIME:SPAN 1E-8"
20 OUTPUT 717;":CALC1:FILT:TIME:SPAN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STARt

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STARt

Description

This command sets/gets the start value of the gate used for the gating function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The start value of the gate
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	-1E-8
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim FilCent As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STARt = 0
FilCent = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STARt
```

Related objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STOP

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Gating > Start

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:STARt <numeric>

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:STARt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:TIME:STAR 0"
20 OUTPUT 717;":CALC1:FILT:TIME:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STATe

Description

This command turns ON/OFF the gating function of the time domain function, for the active trace of selected channel (*Ch*).

You can turn ON the gating function only when the sweep type is the linear sweep and the number of points is 3 or more. If you execute this object to turn ON the gating function when the sweep type is other than linear sweep or the number of points is less than 3, an error occurs and the object is ignored.

When the sweep type is the power sweep, you cannot turn ON the gating function. If you execute this object to turn ON the gating function during the power sweep, an error occurs and the object is ignored.

Variable

Parameter	*Status*
Description	ON/OFF of the gating function
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the gating function. • False or OFF: Turns OFF the gating function.
Preset value	False or OFF

Examples

```
Dim Gating As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STATe = True
Gating = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SENSe(Ch).SWEep.POINts

Equivalent key

Analysis > **Gating** > **Gating**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:STATe {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:TIME:STAT ON"
20 OUTPUT 717;":CALC1:FILT:TIME:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STOP

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.STOP

Description

This command sets/gets the stop value of the gate used for the gating function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The stop value of the gate
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	1E-8
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim FilStop As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STOP = 2E-8
FilStop = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.STOP
```

Related objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STARt

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Gating > Stop

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:STOP <numeric>

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME:STOP?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:TIME:STOP 2E-8"
20 OUTPUT 717;":CALC1:FILT:TIME:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.TYPE

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FILTer.GATE.TIME.TYPE

Description

This command sets/gets the gate type used for the gating function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The gate type
Data type	Character string type (String)
Range	Select from the following. • "BPASs": Specifies the band-pass type. • "NOTCh": Specifies the notch type.
Preset value	"BPASs"

Examples

```
Dim FilType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe = "notc"
FilType = SCPI.CALCulate(1).SELected.FILTer.GATE.TIME.SHAPe
```

Related objects

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.SHAPe

SCPI.CALCulate(Ch).SELected.FILTer.GATE.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > **Gating** > **Type**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME[:TYPE]
{BPASs|NOTCh}

:CALCulate{[1]-4}[:SELected]:FILTer[:GATE]:TIME[:TYPE]?

Query response

{BPAS|NOTC}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FILT:TIME NOTC"
20 OUTPUT 717;":CALC1:FILT:TIME?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FORMat

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FORMat = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FORMat

Description

This command sets/gets the data format of the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Data format
Data type	Character string type (String)
Range	Select from the following: • "MLOGarithmic": Specifies the log magnitude format. • "PHASe": Specifies the phase format. • "GDELay": Specifies the group delay format. • "SLINear": Specifies the Smith chart format (Lin/Phase). • "SLOGarithmic": Specifies the Smith chart format (Log/Phase). • "SCOMplex": Specifies the Smith chart format (Re/Im). • "SMITh": Specifies the Smith chart format (R+jX). • "SADMittance": Specifies the Smith chart format (G+jB). • "PLINear": Specifies the polar format (Lin/Phase). • "PLOGarithmic": Specifies the polar format (Log/Phase). • "POLar": Specifies the polar format (Re/Im).

	• "MLINear": Specifies the linear magnitude format. • "SWR": Specifies the SWR format. • "REAL": Specifies the real format. • "IMAGinary": Specifies the imaginary format. • "UPHase": Specifies the expanded phase format. • "PPHase": Specifies the positive phase format.
Preset value	"MLOGarithmic"

Examples

```
Dim Fmt As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FORMat = "smit"
Fmt = SCPI.CALCulate(1).SELected.FORMat
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Format > **Log Mag**|**Phase**|**Group Delay**|**Lin Mag**|**SWR**|**Real**|**Imaginary**|**Expand Phase**|**Positive Phase**

Format > **Smith** > **Lin/Phase**|**Log/Phase**|**Real/Imag**|**R+jX**|**G+jB**

Format > **Polor** > **Lin/Phase**|**Log/Phase**|**Real/Imag**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FORMat {MLOGarithmic|PHASe|GDELay|SLINear|SLOGarithmic|SCOMplex|SMITh|SADMittance|PLINear|PLOGarithmic|POLar|MLINear|SWR|REAL| IMAGinary|UPHase|PPHase}

:CALCulate{[1]-4}[:SELected]:FORMat?

Query response

{MLOG|PHAS|GDEL|SLIN|SLOG|SCOM|SMIT|SADM|PLIN|PLOG|POL|MLIN|SWR| REAL|IMAG|UPH|PPH}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FORM SLIN"
20 OUTPUT 717;":CALC1:FORM?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DATA

Description

This command reads the analysis result of the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates the array data (analysis result) of N (number of data pairs)×2. N (number of data pairs) can be read out with the SCPI.CALCulate(Ch).SELected.FUNCtion.POINts object. Where n is an integer between 1 and N. • *Data(n×2-2)* :Response value or analysis result of the searched n-th measurement point. • *Data(n×2-1)* :Stimulus value of the searched n-th measurement point. Always 0 for the analysis of the mean value, the standard deviation, and the difference between the maximum value and the minimum value. The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim AnaData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "mean"
SCPI.CALCulate(1).SELected.FUNCtion.EXECute
AnaData = SCPI.CALCulate(1).SELected.FUNCtion.DATA
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

SCPI.CALCulate(Ch).SELected.FUNCtion.POINts

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:DATA?

Query response

<numeric 1>,… ,<numeric N×2><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:POIN?"
20 ENTER 717;A
30 REDIM B(1:2*A)
40 OUTPUT 717;":CALC1:FUNC:DATA?"
50 ENTER 717;B(*)
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.COUPle

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.COUPle = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.COUPle

Description

This command specifies whether to set the coupling of the analysis range of the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object for all traces, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the trace coupling of the analysis range.
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON:Specifies the analysis range with the trace coupling. • False or OFF:Specifies the analysis range for each trace.
Preset value	True or ON

Examples

```
Dim TrCpl As Boolean
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.COUPle = False
TrCpl = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.COUPle
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain:COUPle {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain:COUPle?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:DOM:COUP OFF"
20 OUTPUT 717;":CALC1:FUNC:DOM:COUP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STARt

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STARt

Description

This command sets/gets the start value of the analysis range of the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the selected channel (*Ch*).

When the trace coupling is OFF, the active trace is the target to be set.

Variable

Parameter	*Value*
Description	Start value of the analysis range
Data type	Double precision floating point type (Double)
Preset value	0
Unit	Hz (hertz), dBm or s (second)

Examples

```
Dim AnaStar As Double
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt = 1.5E9
AnaStar = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STOP

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain:STARt <numeric>

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain:STARt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:DOM:STAR 1.7E9"
20 OUTPUT 717;":CALC1:FUNC:DOM:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STATe

Description

This command sets/gets whether to use an arbitrary range when executing the analysis with the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the selected channel (*Ch*).

When the trace coupling is OFF, the active trace is the target to be set.

Variable

Parameter	*Status*
Description	Selection of the analysis range
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Specifies an arbitrary range • False or OFF: Specifies the entire sweep range
Preset value	False or OFF

Examples

```
Dim AnaRnge As Boolean
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STARt = 1.5E9
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 1.8E9
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe = True
AnaRnge = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STATe
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STARt

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STOP

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:DOM ON"
20 OUTPUT 717;":CALC1:FUNC:DOM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STOP

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.DOMain.STOP

Description

This command sets/gets the stop value of the analysis range of the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the selected channel (*Ch*).

When the trace coupling is OFF, the active trace is the target to be set.

Variable

Parameter	*Value*
Description	Stop value of the analysis range
Data type	Double precision floating point type (Double)
Preset value	0
Unit	Hz (hertz), dBm or s (second)

Examples

```
Dim AnaStop As Double
SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP = 1.8E9
AnaStop = SCPI.CALCulate(1).SELected.FUNCtion.DOMain.STOP
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STARt

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.COUPle

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain:STOP <numeric>

:CALCulate{[1]-4}[:SELected]:FUNCtion:DOMain:STOP?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:DOM:STOP 1.8E9"
20 OUTPUT 717;":CALC1:FUNC:DOM:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.EXECute

Object type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.EXECute

Description

This command executes the analysis specified with the SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE object, for the active trace of the selected channel (*Ch*).

Examples

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.EXECute

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.FUNCtion.DOMain.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:EXECute

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:EXEC"
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PEXCursion

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PEXCursion = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.PEXCursion

Description

This command sets/gets the lower limit of peak excursion value (the minimum value of the difference relative to the right and left adjacent measurement points) when executing the peak search with the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Lower limit of peak excursion value
Data type	Double precision floating point type (Double)
Range	0 to 5E8
Preset value	3
Unit	Varies depending on the data format. • Log magnitude (MLOG) : dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH) : ° (degree) • Group delay (GDEL) : s (second) • Others : No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PeakExc As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion = 1.5
PeakExc = SCPI.CALCulate(1).SELected.FUNCtion.PEXCursion
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.FUNCtion.PPOLarity

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:PEXCursion <numeric>

:CALCulate{[1]-4}[:SELected]:FUNCtion:PEXCursion?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:PEXC 0.2"
20 OUTPUT 717;":CALC1:FUNC:PEXC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.POINts

Object type

Property (**Read Only**)

Syntax

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.POINts

Description

This command reads the number of data pairs of the analysis result of the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the active trace of selected channel (*Ch*).

For the analysis of the mean value or the search of the maximum value, 1 is always read out; for the search of all peaks or the search of all targets, the total number of searched measurement points is read out.

Variable

Parameter	*Value*
Description	Number of analyzed data pairs
Data type	Long integer type (Long)
Preset Value	0

Examples

```
Dim AnaPoin As Long
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "ape"
SCPI.CALCulate(1).SELected.FUNCtion.EXECute
AnaPoin = SCPI.CALCulate(1).SELected.FUNCtion.POINts
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:POINts?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:POIN?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PPOLarity

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.PPOLarity = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FUNCtion.PPOLarity

Description

This command sets/gets the polarity when performing the peak search with the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Polarity for peak search
Data type	Character string type (String)
Range	Select from the following: • "POSitive": Specifies the positive peak. • "NEGative": Specifies the negative peak. • "BOTH": Specifies both the positive peak and the negative peak.
Preset value	"POSitive"

Examples

```
Dim PeakPol As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity = "both"
PeakPol = SCPI.CALCulate(1).SELected.FUNCtion.PPOLarity
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.FUNCtion.PEXCursion

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:PPOLarity {POSitive|NEGative|BOTH}

:CALCulate{[1]-4}[:SELected]:FUNCtion:PPOLarity?

Query response

{POS|NEG|BOTH}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:PPOL BOTH"
20 OUTPUT 717;":CALC1:FUNC:PPOL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TARGet

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TARGet = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.FUNCtion.TARGet

Description

This command sets/gets the target value when performing the target search with the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Target value
Data type	Double precision floating point type (Double)
Range	-5E8 to 5E8
Preset value	0
Unit	Varies depending on the data format. • Log magnitude (MLOG) : dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH) : ° (degree) • Group delay (GDEL) : s (second) • Others : No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim TargVal As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "atar"
```

```
SCPI.CALCulate(1).SELected.FUNCtion.TARGet = -12.5
TargVal = SCPI.CALCulate(1).SELected.FUNCtion.TARGet
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.FUNCtion.TTRansition

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:TARGet <numeric>

:CALCulate{[1]-4}[:SELected]:FUNCtion:TARGet?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:TARG -12.5"
20 OUTPUT 717;":CALC1:FUNC:TARG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TTRansition = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FUNCtion.TTRansition

Description

This command sets/gets the transition type when performing the target search with the SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute object, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Transition type for search
Data type	Character string type (String)
Range	Select from the following: • "POSitive": Specifies the positive transition. • "NEGative": Specifies the negative transition. • "BOTH": Specifies both the positive transition and the negative transition.
Preset value	"BOTH"

Examples

```
Dim TargTran As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "atar"
SCPI.CALCulate(1).SELected.FUNCtion.TTRansition = "pos"
TargTran = SCPI.CALCulate(1).SELected.FUNCtion.TTRansition
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.FUNCtion.TARGet

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:TTRansition {POSitive|NEGative|BOTH}

:CALCulate{[1]-4}[:SELected]:FUNCtion:TTRansition?

Query response

{POS|NEG|BOTH}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:TTR NEG"
20 OUTPUT 717;":CALC1:FUNC:TTR?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TYPE

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.FUNCtion.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.FUNCtion.TYPE

Description

This command sets/gets the type of analysis, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Analysis type
Data type	Character string type (String)
Range	Select from the following: • "PTPeak": Specifies the analysis of the difference between the maximum value and the minimum value (Peak to Peak). • "STDEV": Specifies the analysis of the standard deviation. • "MEAN": Specifies the analysis of the mean value. • "MAXimum": Specifies the search for the maximum value. • "MINimum": Specifies the search for the minimum value. • "PEAK": Specifies the search for the peak. • "APEak": Specifies the search for all peaks. • "ATARget": Specifies the search for all targets. • "SDEViation": Specifies the search for all deviations.
Preset	"PTPeak"

value	

Examples

```
Dim AnaType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.FUNCtion.TYPE = "atar"
AnaType = SCPI.CALCulate(1).SELected.FUNCtion.TYPE
```

Related objects

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.FUNCtion.PEXCursion

SCPI.CALCulate(Ch).SELected.FUNCtion.PPOLarity

SCPI.CALCulate(Ch).SELected.FUNCtion.TARGet

SCPI.CALCulate(Ch).SELected.FUNCtion.TTRansition

SCPI.CALCulate(Ch).SELected.FUNCtion.EXECute

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:FUNCtion:TYPE {PTPeak| STDEV|MEAN| MAXimum|MINimum|PEAK|APEak|ATARget|SDEViation}

:CALCulate{[1]-4}[:SELected]:FUNCtion:TYPE?

Query response

{PTP|STDEV|MEAN|MAX|MIN|PEAK|APE|ATAR|SDEV}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:FUNC:TYPE PEAK"
20 OUTPUT 717;":CALC1:FUNC:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.LIMit.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.LIMit.DATA = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.LIMit.DATA

Description

This command sets/gets the limit table for the limit test, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates the array data (for limit line) of 1 + Num (number of limit lines) × 5. Where n is an integer between 1 and Num. • *Data(0)* :The number of limit lines you want to set. Specify an integer ranging 0 to 100. When the number of limit lines is set to 0 (clears the limit table), the variable Data is only required with *Data(0)*. • *Data(n×5-4)* :The type of the n-th line. Specify an integer 0 to 2 as follows. 0: OFF 1: Upper limit line 2: Lower limit line • *Data(n×5-3)* :The value on the horizontal axis (frequency/power/time) of the start point of the n-th line. • *Data(n×5-2)* :The value on the horizontal axis (frequency/power/time) of the end point of the n-th line. • *Data(n×5-1)* :The value on the vertical axis of the start point of the n-th line. • *Data(n×5)* :The value on the vertical axis of the end point of the n-th line. The index of the array starts from 0.
Data type	Variant type (Variant)

Note	If there is no array data, an error occurs at execution and the object is ignored. For *Data(n×5-4)* in the array data, if you specify an integer other than 0, 1 or 2, an error occurs at execution. For *Data(n×5-3)*, *Data(n×5-2)*, *Data(n×5-1)*, and *Data(n×5)* in the array data, if the specified value is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim LimData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = Array(1,1,1e6,1e9,0,0)
LimData = SCPI.CALCulate(1).SELected.LIMit.DATA
```

```
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = Array(0) 'Clear Limit Table
```

```
Dim LimData(5) As Variant
Dim Ref As Variant
LimData(0) = 1
LimData(1) = 1
LimData(2) = 1e6
LimData(3) = 1e9
LimData(4) = 0
LimData(5) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = LimData
Ref = SCPI.CALCulate(1).SELected.LIMit.DATA
```

```
Dim LimData(0) As Variant
LimData(0) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.DATA = LimData 'Clear Limit Table
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI.CALCulate(Ch).SELected.LIMit.DISPlay.STATe

Equivalent key

Analysis > Limit Test > Edit Limit Line

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:DATA <numeric 1>, ... ,<numeric 1+(N×5)>

:CALCulate{[1]-4}[:SELected]:LIMit:DATA?

Query response

{numeric 1}, ... ,{numeric 1+(N×5)}<newline><^END>

Example of use

```
10 DIM B(1:2,1:5)
20 OUTPUT 717;":CALC1:LIM:DATA 2,1,1E9,3E9,0,0,2,1E9,3E9,-3,-3"
30 OUTPUT 717;":CALC1:LIM:DATA?"
40 ENTER 717;A,B(*)
```

```
10 OUTPUT 717;":CALC1:LIM:DATA 0" ! Clear Limit Table
```

SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.CLIP

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.CLIP = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.CLIP

Description

This command sets/gets whether to display the part of the limit line that is not used for evaluation, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Displays the clipped limit lines
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Displays the clipped limit lines. • False or OFF: Displays the entire limit lines.
Preset Value	True or ON

Examples

```
Dim LimClip as Boolean
LimClip = True
SCPI.CALCulate(4).SELected.LIMit.DISPlay.CLIP = LimClip
LimClip = SCPI.CALCulate(4).SELected.LIMit.DISPlay.CLIP
```

Related Objects

SCPI.CALCulate(Ch).SELected.LIMit.DISPlay.STATe

Equivalent Key

Analysis > **Limit Test** > **Clip Lines**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:DISPlay:CLIP {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:LIMit:DISPlay:CLIP?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:LIM:DISP:CLIP ON"
20 OUTPUT 717;":CALC1:LIM:DISP:CLIP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.LIMit.DISPlay.STATe

Description

This command turns ON/OFF the limit line display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Limit line display
Data type	Boolean type (Boolean)
Range	Select from the following. • True or ON: Turns ON the limit line display. • False or OFF: Turns OFF the limit line display.
Preset value	False or OFF

Examples

```
Dim LimDisp As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit = True
LimDisp = SCPI.CALCulate(1).SELected.LIMit.DISPlay.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

Equivalent key

Analysis > **Limit Test** > **Limit Line**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:DISPlay[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:LIMit:DISPlay[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:LIM:DISP ON"
20 OUTPUT 717;":CALC1:LIM:DISP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.FAIL

Object type

Property (**Read Only**)

Syntax

Status = SCPI.CALCulate(*Ch*).SELected.LIMit.FAIL

Description

This command reads out the limit test result, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Limit test result
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: The limit test result is FAIL. • False or OFF: The limit test result is PASS.
Note	When the limit test is set to OFF, False or OFF is always read out.

Examples

```
Dim Result As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
Result = SCPI.CALCulate(1).SELected.LIMit.FAIL
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:FAIL?

Query response

{1|0}<newline><^END>

	Description
1	The limit test result is FAIL.
0	The limit test result is PASS.

When the limit test is set to OFF, 0 is always read out.

Example of use

```
10 OUTPUT 717;":CALC1:LIM:FAIL?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.OFFSet.AMPLitude

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude = *Value*

Value = SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude

Description

This command sets/gets the limit line offset of response for the active trace of selected channel (*Ch*). The setting of the limit line does not change even if the offset value is changed.

Variable

Parameter	*Value*
Description	The limit line offset of Response (Vertical offset)
Data type	Double precision floating point type (Double)
Range	-5E8 to 5E8
Preset value	0
Unit	Varies depending on the data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): ° (degree) • Group delay (GDEL): s (second) • Others: No unit

Examples

```
Dim LimOffset As Double
SCPI.CALCulate(1).SELected.LIMit.OFFSet.AMPLitude = -10
LimOffset = SCPI.CALCulate(1).SELected.LIMit.OFFSet.AMPLitude
```

Related objects

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.MARKer

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus

Equivalent key

Analysis > **Limit Test** > **Limit Line Offsets** > **Amplitude Offset**

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:LIMit:OFFSet:AMPLitude <numeric>
:CALCulate{[1]-4}[:SELected]:LIMit:OFFSet:AMPLitude?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:LIM:OFFS:AMPL -10"
20 OUTPUT 717;":CALC1:LIM:OFFS:AMPL?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.OFFSet.MARKer

Object type

Method (**Write-only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.LIMit.OFFSet.MARKer

Description

This command sets the active marker value to amplitude offset using the limit line for the active trace of selected channel (*Ch*). The setting of the limit line does not change even if the offset value is changed. When the markers are not displayed, this command does not operate.

Variable

None

Examples

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.OFFSet.MARKer

Related objects

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.STIMulus

Equivalent key

Analysis > **Limit Test** > **Limit Line Offsets** > **Marker -> Amplitude Offset**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:OFFSet:MARKer

Example of use

10 OUTPUT 717;":CALC1:LIM:OFFS:MARK"

SCPI.CALCulate(*Ch*).SELected.LIMit.OFFSet.STIMulus

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.LIMit.OFFSet.STIMulus = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.LIMit.OFFSet.STIMulus

Description

This command sets/gets the stimulus offset of the limit line, for the active trace of the selected channel (*Ch*).

NOTE The setting of the limit line doesn't change even if the offset value is changed.

Variable

Parameter	*Value*
Description	The stimulus offset of the limit line
Data type	Double precision floating point type (Double)
Range	-1E12 to 1E12
Preset value	0
Unit	Hz (hertz), dBm or second

Examples

```
Dim LimOffset As Double
SCPI.CALCulate(1).SELected.LIMit.OFFSet.STIMulus = 1E9
LimOffset = SCPI.CALCulate(1).SELected.LIMit.OFFSet.STIMulus
```

Related objects

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.AMPLitude

SCPI.CALCulate(Ch).SELected.LIMit.OFFSet.MARKer

Equivalent key

Analysis > **Limit Test** > **Limit Line Offsets** > **Stimulus Offset**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:OFFSet:STIMulus <numeric>

:CALCulate{[1]-4}[:SELected]:LIMit:OFFSet:STIMulus?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:LIM:OFFS:STIM 5E3"
20 OUTPUT 717;":CALC1:LIM:OFFS:STIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.ALL

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.ALL

Description

This command reads the bandwidth test results (stimulus value, limit test result, upper limit value and lower limit value of all measurement points), for the active trace of selected channel (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command).

Variable

Parameter	*Data*
Description	Indicates the array data (for limit line) of NOP (number of measurement points) × 4. Where n is an integer between 1 and NOP. • *Data(n×4-3)* The stimulus value for the measurement point. • *Data(n×4-2)* The limit test result. Specify an integer -1 to 1 as follows. -1: No limit 0: Fail 1: Pass • *Data(n×4-1)* The upper limit value at the measurement point. (If there is no limit at this point, reads out the 0.) • *Data(n×4)* The lower limit value at the measurement point. (If there is no limit at this point, reads out the 0.) The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim LimData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
LimData = SCPI.CALCulate(1).SELected.LIMit.REPort.ALL
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI.CALCulate(Ch).SELected.LIMit.REPort.DATA

SCPI.CALCulate(Ch).SELected.LIMit.REPort.POINts

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:REPort:ALL?

Example of use

```
10 OUTPUT 717;":SENS1:SWE:POIN?"
20 ENTER 717;A
30 REDIM B(1:4*A)
40 OUTPUT 717;":CALC1:LIM:REP:ALL?"
50 ENTER 717;B(*)
```

SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.DATA

Description

This command reads the stimulus values (frequency, power level or time) at all the measurement points that failed the limit test, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates the array data for failed measurement points (can be read out with the SCPI.CALCulate(Ch).SELected.LIMit.REPort.POINts object).
Data type	Variant type (Variant)

Examples

```
Dim FailData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
FailData = SCPI.CALCulate(1).SELected.LIMit.REPort.DATA
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.REPort.POINts

SCPI.CALCulate(Ch).SELected.LIMit.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:REPort[:DATA]?

Query response

{numeric 1},… ,{numeric N}<newline><^END>

Where N is the number of the measurement points that failed (can be read out with the :CALC{1-16}:LIM:REP:POIN? command).

Example of use

```
10 OUTPUT 717;":CALC1:LIM:REP:POIN?"
20 ENTER 717;A
30 REDIM B(1:A)
40 OUTPUT 717;":CALC1:LIM:REP?"
50 ENTER 717;B(*)
```

SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.POINts

Object type

Property (**Read Only**)

Syntax

Value = SCPI.CALCulate(*Ch*).SELected.LIMit.REPort.POINts

Description

This command reads the number of the measurement points that failed the limit test, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of measurement points that failed
Data type	Long integer type (Long)
Preset Value	0

Examples

```
Dim FailPoin As Long
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
FailPoin = SCPI.CALCulate(1).SELected.LIMit.REPort.POINts
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit:REPort:POINts?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:LIM:REP:POIN?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.LIMit.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.LIMit.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.LIMit.STATe

Description

This command turns ON/OFF the limit test function, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the limit test function
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the limit test function. • False or OFF: Turns OFF the limit test function.
Preset value	False or OFF

Examples

```
Dim LimTest As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.LIMit.STATe = True
LimTest = SCPI.CALCulate(1).SELected.LIMit.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.LIMit.DISPlay.STATe

SCPI.DISPlay.FSIGn

Equivalent key

Analysis > **Limit Test** > **Limit Test**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:LIMit[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:LIMit[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:LIM ON"
20 OUTPUT 717;":CALC1:LIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).ACTivate

Object type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).ACTivate

Description

This command sets the marker 1 to 9 (*Mk*) and reference marker (*Mk*:10) to the active marker, for the active trace of selected channel (*Ch*).

NOTE If you set a marker as not to be displayed to the active marker, the marker display is automatically set to ON.

Variable

Parameter	*Mk*
Description	Marker number
Data type	Long integer type (Long)
Range	1 to 10 Notice that 10 is for the reference marker.
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).ACTivate

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.DISPlay.WINDow(Ch).ACTivate

Equivalent key

Marker > **Marker 1**|**Marker 2**|**Marker 3**|**Marker 4**|**Ref Marker**

Marker > **More Markers** > **Marker 5**|**Marker 6**|**Marker 7**|**Marker 8**|**Marker 9**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:ACTivate

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:ACT"
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.DATA

Description

This command reads the bandwidth search result of marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of selected channel (*Ch*).

If the bandwidth search is impossible, an error occurs when executed and the object is ignored.

Variable

Parameter	*Data*
Description	Indicates 4-element array data (bandwidth search result). • *Data(0)* :The bandwidth. • *Data(1)* :Center point frequency of the 2 cutoff frequency points. • *Data(2)* :The Q value. • *Data(3)* :Insertion loss The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim BandData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
BandData = SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.DATA
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer.BWIDth.STATe

SCPI.CALCulate(Ch).SELected.MARKer(Mk).BWIDth.THReshold

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:BWIDth:DATA?

Query Response

{numeric1},{numeric2},{numeric3},{numeric4},<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:BWID:DATA?"
20 ENTER 717;A,B,C,D
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.THReshold

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.THReshold = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).BWIDth.THReshold

Description

This command sets/gets the bandwidth definition value (the value to define the pass-band of the filter) of marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Bandwidth definition value (the value to define the pass band of the filter)
Data type	Double precision floating point type (Double)
Range	-5E8 to 5E8
Preset value	-3
Unit	Varies depending on the data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): ° (degree) • Group delay (GDEL): s (second) • Others: No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim BandVal As Double
SCPI.CALCulate(1).PARameter(1).SELect
```

```
SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.THReshold = -6
BandVal = SCPI.CALCulate(1).SELected.MARKer(1).BWIDth.THReshold
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer.BWIDth.STATe

Equivalent key
Marker Search > Bandwidth Value
Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:BWIDth:THReshold <numeric>

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:BWIDth:THReshold?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:BWID:THR 6"
20 OUTPUT 717;":CALC1:MARK1:BWID:THR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.EXECute

Object type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.EXECute

Description

This command executes search with marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*).

To specify the type of the search, use the SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion. TYPE object. (No read)

Examples

```
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "maximum"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.EXECute
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STATe

Equivalent key

Marker Search > **Max**|**Min**

Marker Search > **Peak** > **Search Peak**|**Search Left**|**Search Right**

Marker Search > **Target** > **Search Target**|**Search Left**|**Search Right**

When performing the operation from the front panel, you select the search type and execute the search at the same time.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:EXECute

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:EXEC"
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PEXCursion

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PEXCursion = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PEXCursion

Description

This command sets/gets the lower limit of peak excursion value when executing the peak search with marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of selected channel (*Ch*). Peak excursion value is the minimum value of the difference relative to the right and left adjacent measurement points.

Variable

Parameter	*Value*
Description	Lower limit of peak excursion value
Data type	Double precision floating point type (Double)
Range	0 to 5E8
Preset value	3
Unit	Varies depending on the data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): × (degree) • Group delay (GDEL): s (second) • Others: No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PeakExc As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PEXCursion = 0.2
PeakExc = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PEXCursion
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PPOLarity

Equivalent key

Marker Search > **Peak** > **Peak Excursion**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:PEXCursion <numeric>

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:PEXCursion?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:PEXC 0.2"
20 OUTPUT 717;":CALC1:MARK1:FUNC:PEXC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PPOLarity

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PPOLarity = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.PPOLarity

Description

This comamnd set/get the polarity of the peak search with marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Polarity for peak search
Data type	Character string type (String)
Range	Select from the following: • "POSitive" Specifies the positive peak. • "NEGative" Specifies the negative peak. • "BOTH" Specifies both the positive peak and the negative peak.
Preset value	"POSitive"

Examples

```
Dim PeakPol As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "peak"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PPOLarity = "both"
PeakPol = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.PPOLarity
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PEXCursion

Equivalent key

Marker Search > **Peak** > **Peak Polarity** > **Positive**|**Negative**|**Both**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:PPOLarity {POSitive|NEGative|BOTH}

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:PPOLarity?

Query response

{POS|NEG|BOTH}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:PPOL NEG"
20 OUTPUT 717;":CALC1:MARK1:FUNC:PPOL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TARGet

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TARGet = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TARGet

Description

This command sets/gets the target value to be searched with marker 1 to 9 (*Mk*) and reference marker (*Mk*:10, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Target value for target search
Data type	Double precision floating point type (Double)
Range	-5E8 to 5E8
Preset value	0
Unit	Varies depending on the data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): ° (degree) • Group delay (GDEL): s (second) • Others: No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim TargVal As Double
SCPI.CALCulate(1).PARameter(1).SELect
```

```
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TARGet = -12.5
TargVal = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TARGet
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TTRansition

Equivalent key

Marker Search > **Target** > **Target Value**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:TARGet <numeric>

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:TARGet?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TARG -12.5"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TARG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TRACking

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TRACking = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TRACking

Description

This command turns ON/OFF the search tracking (function to repeat search for each sweep) for marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the marker search tracing
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the search tracking. • False or OFF: Turns OFF the search tracking.
Preset value	False or OFF

Examples

```
Dim SrchTrac As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "targ"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TRACking = True
SrchTrac = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TRACking
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

Equivalent key

Marker Search > **Tracking**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:TRACking {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:TRACking?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TRAC ON"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TRAC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TTRansition = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TTRansition

Description

This command selects the transition type of the target search, for marker 1 to 9 (*Mk*) and reference marker (*Mk*:10) of the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Transition type for search
Data type	Character string type (String)
Range	Select from the following: • "POSitive": Specifies the positive transition. • "NEGative": Specifies the negative transition. • "BOTH": Specifies both the positive transition and the negative transition.
Preset value	"BOTH"

Examples

```
Dim TargTran As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "targ"
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TTRansition = "neg"
TargTran = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TTRansition
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TYPE

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TARGet

Equivalent key

Marker Search > **Target** > **Target Transition** > **Positive**|**Negative**|**Both**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:TTRansition {POSitive| NEGative|BOTH}

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion:TTRansition?

Query response

{POS|NEG|BOTH}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TTR NEG"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TTR?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TYPE

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).FUNCtion.TYPE

Description

This command selects the search type for marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Search type of marker
Data type	Character string type (String)
Range	Select from the following: • "MAXimum": Sets the search type to the maximum value. • "MINimum": Sets the search type to the minimum value. • "PEAK": Sets the search type to the peak search. • "LPEak": Sets the search type to the peak search to the left from the marker position. • "RPEak": Sets the search type to the peak search to the right from the marker position. • "TARGet": Sets the search type to the target search. • "LTARget": Sets the search type to the target search to the left from the marker position. • "RTARget": Sets the search type to the target search to the right from the marker position.
Preset value	"MAXimum"

Examples

```
Dim SrchType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE = "targ"
SrchType = SCPI.CALCulate(1).SELected.MARKer(1).FUNCtion.TYPE
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PEXCursion

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.PPOLarity

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TARGet

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.TTRansition

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

Equivalent key

Marker Search > **Max**|**Min**

Marker Search > **Peak** > **Search Peak**|**Search Left**|**Search Right**

Marker Search > **Target** > **Search Target**|**Search Left**|**Search Right**

NOTE When performing the operation from the front panel, you select the search type and execute the search at the same time.

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion
:TYPE {MAXimum|
MINimum|PEAK|LPEak|RPEak|TARGet|LTARget|RTARget}
:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:FUNCtion
:TYPE?
```

Query response

{MAX|MIN|PEAK|LPE|RPE|TARG|LTAR|RTAR}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:FUNC:TYPE PEAK"
20 OUTPUT 717;":CALC1:MARK1:FUNC:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).NOTCh.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.MARKer(Mk).NOTCh.DATA

Description

This command reads the notch search result of marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*).

If the notch search is impossible, an error occurs and the command is ignored. In this case, no query response is obtained.

Variable

Parameter	*Data*
Description	Indicates 4-element array data (notch bandwidth search result). • *Data(0)* :The bandwidth. • *Data(1)* :Center point frequency of the 2 cutoff frequency points. • *Data(2)* :The Q value. • *Data(3)* :Insertion loss The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim NotchData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
BandData = SCPI.CALCulate(1).SELected.MARKer(1).NOTCH.DATA
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.NOTCh.STATe

SCPI.CALCulate(Ch).SELected.MARKer(Mk).NOTCh.THReshold

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:NOTCh:DATA?

Query response

{value 1},{value 2},{value 3},{value 4}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:NOTC :DATA?"
20 ENTER 717;A,B,C,D
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).NOTCh.THReshold

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).NOTCh.THReshold = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).NOTCh.THReshold

Description

This command sets/gets the notch definition value of marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*), specified with the :CALC{[1]-4}:PAR{[1]-4}:SEL command.

Variable

Parameter	*Value*
Description	Notch definition value
Range	-5E8 to 5E8
Preset Value	-3
Unit	Varies depending on the data format as follows: • Amplitude (MLOG):dB (decibel) • Phase (PHAS), Expanded phase (UPH),Positive phase (PPH): ° (degree) • Group delay (GDEL): s (second) • Others: No unit

NOTE If the specified parameter is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim NotchVal As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).NOTCH.THReshold = -6
NotchVal = SCPI.CALCulate(1).SELected.MARKer(1).NOTCH.THReshold
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.NOTCh.STATe

SCPI.CALCulate(Ch).SELected.MARKer(Mk).NOTCh.DATA

Equivalent key

Marker Search > **Notch Value**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:NOTCh:THReshold <value>

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:NOTCh:THReshold?

Query response

{value}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:NOTC :THR 5"
20 OUTPUT 717;":CALC1:MARK1:NOTC :THR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).SET

Object type

Property (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).SET = *Param*

Description

This command sets the value at the position of marker 1 to 9 (*Mk*) and reference marker (*Mk*:10) to the value of the instrument setting item (*Param*), for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Instrument setting item
Data type	Character string type (String)
Range	Select from the following: • "STARt": Sets the sweep start value to the stimulus value at the marker position. • "STOP": Sets the sweep stop value to the stimulus value at the marker position. • "CENTer": Sets the sweep center value to the stimulus value at the marker position. • "RLEVel": Sets the reference line value to the response value at the marker position. • "DELay": Sets the electrical delay time value to the value of the group delay at the marker position (a value smoothed with the aperture of 20%).

Examples

```
Dim MkrTo As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).SET = "cent"
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe

Equivalent key

Marker Fctn > Marker -> Start|Marker -> Stop|Marker -> Center|Marker -> Reference|Marker -> Delay

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:SET {STARt|
STOP|CENTer|RLEVel|DELay}
```

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:SET CENT"
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).STATe

Description

This command turns ON/OFF the display of marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of display of markers 1 to 9 and reference marker
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the display of the marker. • False or OFF: Turns OFF the display of the marker.
Preset value	False or OFF

Examples

```
Dim Mkr As Boolean
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.MARKer(10).STATe = True
Mkr = SCPI.CALCulate(1).SELected.MARKer(10).STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

When turning ON the display of the marker

Marker > **Marker 1**|**Marker 2**|**Marker 3**|**Marker 4**|**Ref Marker**

Marker > **More Markers** > **Marker 5**|**Marker 6**|**Marker 7**|**Marker 8**|**Marker 9**

When performing the operation from the front panel, a marker set to ON is automatically set to the *active marker* .

When turning OFF the display of the marker

Marker > **Clear Marker Menu** > **Marker 1**|**Marker 2**|**Marker 3**|**Marker 4**|**Marker 5**|**Marker 6**|**Marker 7**|**Marker 8**|**Marker 9**|**Ref Marker**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1 ON"
20 OUTPUT 717;":CALC1:MARK1?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).X

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).X = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).X

Description

This command set the stimulus value for marker 1 to 9 (*Mk*) and reference marker (*Ch*:10), for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Stimulus value of the marker
Data type	Double precision floating point type (Double)
Range	Sweep start value to sweep stop value
Preset value	Sweep start value
Unit	Hz (hertz), dBm or s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim MkrX As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).X = 1E9
MkrX = SCPI.CALCulate(1).SELected.MARKer(1).X
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe

SCPI.CALCulate(Ch).SELected.MARKer(Mk).Y

Equivalent key

Marker > **Marker 1**|**Marker 2**|**Marker 3**|**Marker 4**|**Ref Marker**

Marker > **More Markers** > **Marker 5**|**Marker 6**|**Marker 7**|**Marker 8**|**Marker 9**

When performing the operation from the front panel, you turn ON the marker and set the stimulus value at the same time.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:X <numeric>

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:X?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:X 1E9"
20 OUTPUT 717;":CALC1:MARK1:X?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).Y

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.MARKer(*Mk*).Y

Description

This command reads the response value of marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of selected channel (*Ch*).

When the reference marker mode is ON ("True" is specified with the SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe object), the readout value is the value relative to the reference marker.

Variable

Parameter	*Data*
Description	Indicates 2-element array data (response value of marker). • *Data(0)* :Response value (primary value) at the marker position. • *Data(1)* :Response value (secondary value) at the marker position. Always 0 when the data format is not the Smith chart format or the polar format. The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim MkrY As Variant
SCPI.CALCulate(1).PARameter(1).SELect
MkrY = SCPI.CALCulate(1).SELected.MARKer(1).Y
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer.REFerence.STATe

SCPI.CALCulate(Ch).SELected.MARKer(Mk).X

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer{[1]-10}:Y?

Query response

{numeric 1},{numeric 2}<newline><^END>

	Description
{numeric 1}	Response value (primary value) at the marker position.
{numeric 2}	Response value (secondary value) at the marker position. Always 0 when the data format is not the Smith chart format or the polar format.

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:Y?"
30 ENTER 717;A,B
```

SCPI.CALCulate(*Ch*).SELected.MARKer.AOFF

Object Type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.AOFF

Description

This command turns OFF all markers for the selected channel (*Ch*).

Examples

SCPI.CALCulate(4).SELected.MARKer.AOFF

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer(Mk).STATe

Equivalent Key

Marker > **Clear Marker Menu** > **All OFF**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:AOFF

Example of use

```
10 OUTPUT 717;":CALC1:MARK:AOFF"
```

SCPI.CALCulate(*Ch*).SELected.MARKer.BWIDth.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.BWIDth.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.BWIDth.STATe

Description

This command turns ON/OFF the bandwidth search result display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the bandwidth search result display
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the bandwidth search result display. • False or OFF: Turns OFF the bandwidth search result display.
Preset value	False or OFF

Examples

```
Dim BandSrch As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer.BWIDth.STATe = True
BandSrch = SCPI.CALCulate(1).SELected.MARKer.BWIDth.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).BWIDth.DATA

SCPI.CALCulate(Ch).SELected.MARKer(Mk).BWIDth.THReshold

Equivalent key

Marker Search > **Bandwidth**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:BWIDth[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:BWIDth[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:BWID ON"
20 OUTPUT 717;":CALC1:MARK:BWID?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.COUPle

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.COUPle = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.COUPle

Description

This command turns ON/OFF the marker coupling between traces, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the marker coupling between traces
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the marker coupling. • False or OFF: Turns OFF the marker coupling.
Preset value	True or ON

Examples

```
Dim MkrCpl As Boolean
SCPI.CALCulate(1).SELected.MARKer.COUPle = False
MkrCpl = SCPI.CALCulate(1).SELected.MARKer.COUPle
```

Equivalent key

Marker Fctn > **Couple**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:COUPle {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:COUPle?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:COUP OFF"
20 OUTPUT 717;":CALC1:MARK:COUP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.DISCrete

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.DISCrete = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.DISCrete

Description

This command turns ON/OFF the discrete mode (mode in which the marker moves only at the measurement points) with marker 1 to 9 (*Mk*) and reference marker (*Mk*:10), for the active trace of selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the marker discrete mode
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the discrete mode. • False or OFF: Turns OFF the discrete mode.
Preset value	False or OFF

Examples

```
Dim MkrDsc As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer(1).DISCrete = True
MkrDsc = SCPI.CALCulate(1).SELected.MARKer(1).DISCrete
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Marker Fctn > **Discrete**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:DISCrete {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:DISCrete?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:DISC OFF"
20 OUTPUT 717;":CALC1:MARK:DISC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.COUPle

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.COUPle = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.COUPle

Description

This command sets/gets the coupling of the marker search range for all traces, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the trace coupling of the marker search range.
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Specifies the search range with the trace coupling. • False or OFF :Specifies the search range for each trace.
Preset value	True or ON

Examples

```
Dim TrCpl As Boolean
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.COUPle = False
TrCpl = SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.COUPle
```

Related objects

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

Equivalent key

Marker Search > **Search Range** > **Couple**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain:COUPle {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain:COUPle?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM:COUP OFF"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM:COUP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STARt

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STARt

Description

This command sets/gets the start value of the marker search range, for the selected channel (*Ch*).

NOTE When the trace coupling is OFF, the active trace is the target to be set.

Variable

Parameter	*Value*
Description	The start value of the search range
Data type	Double precision floating point type (Double)
Preset value	0
Unit	Hz (hertz), dBm or s (second)

Examples

```
Dim SchStar As Double
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STARt = 1.7E9
SchStar = SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STARt
```

Related objects

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STOP

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STATe

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

Equivalent key

Marker Search > **Search Range** > **Start**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain:STARt <numeric>

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain:STARt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STAR 1.7E9"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STATe

Description

This command sets/gets whether to use an arbitrary range when executing the marker search, for the selected channel (*Ch*).

NOTE When the trace coupling is OFF, the active trace is the target to be set.

Variable

Parameter	*Status*
Description	Selects the search range.
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Specifies an arbitrary range. • False or OFF: Specifies the entire sweep range.
Preset value	False or OFF

Examples

```
Dim SchRnge As Boolean
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STARt = 1.5E9
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STOP = 1.8E9
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STATe = True
SchRnge = SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STATe
```

Related objects

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STARt

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STOP

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

Equivalent key

Marker Search > Search Range > Search Range [ON/OFF]

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain[:STATe]
{ON|OFF|1|0}
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain[:STATe]?
```

Query response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM ON"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STOP

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.DOMain.STOP

Description

This command sets/gets the stop value of the marker search range, for the selected channel (*Ch*).

NOTE When the trace coupling is OFF, the active trace is the target to be set.

Variable

Parameter	*Value*
Description	Stop value of the search range
Data type	Double precision floating point type (Double)
Preset value	0
Unit	Hz (hertz), dBm or s (second)

Examples

```
Dim SchStop As Double
SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STOP = 1.8E9
SchStop = SCPI.CALCulate(1).SELected.MARKer.FUNCtion.DOMain.STOP
```

Related objects

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STARt

SCPI.CALCulate(Ch).SELected.MARKer.FUNCtion.DOMain.STATe

SCPI.CALCulate(Ch).SELected.MARKer(Mk).FUNCtion.EXECute

Equivalent key

Marker Search > **Search Range** > **Stop**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain:STOP <numeric>

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:DOMain:STOP?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STOP 1.8E9"
20 OUTPUT 717;":CALC1:MARK:FUNC:DOM:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.PEXCursion

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.PEXCursion = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.PEXCursion

Description

This command sets/gets the lower limit of peak excursion value for the selected channel (*Ch*) when executing the multi peak search.

Variable

Parameter	*Value*
Description	Lower limit of peak excursion value
Data Type	Double precision floating point type (Double)
Range	0 to 5E8
Preset Value	3
Unit	Varies depending on the data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): × (degree) • Group delay (GDEL): s (second) • Others: No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PeakExc as Double

SCPI.CALCulate4.SELected.MARKer.FUNCtion.MULTi.PEXCursion = 0.2
PeakExc = SCPI.CALCulate4.SELected.MARKer.FUNCtion.MULTi.PEXCursion
```

Related Objects
Equivalent Key

Marker Search > Multi Peak > Peak Excursion

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:PEXCursion
<numeric>
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:PEXCursion?
```

Query Response

```
{numeric} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:MULT:PEXC 0"
20 OUTPUT 717;":CALC1:MARK:FUNC:MULT:PEXC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.PPOLarity

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.PPOLarity = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.PPOLarity

Description

This command sets/gets the polarity of the multi peak search, for the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Polarity for multi peak search
Data Type	Character string type (String)
Range	Select from the following: • "POSitive": Specifies the positive peak. • "NEGative": Specifies the negative peak. • "BOTH": Specifies both the positive peak and the negative peak.
Preset Value	"POSitive"

Examples

```
Dim PeakPol as String
SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.PPOLarity = "both"
PeakPol = SCPI.CALCulate4.SELected.MARKer.FUNCtion.MULTi.PPOLarity
```

Related Objects

Equivalent Key

Marker Search > **Multi Peak** > **Peak Polarity** > **Positive**|**Negative**|**Both**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:PPOLarity {POSitive|NEGative|BOTH}

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:PPOLarity?

Query Response

{POS|NEG|BOTH} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:MULT:PPOL NEG"
20 OUTPUT 717;":CALC1:MARK:FUNC:MULT:PPOL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TARGet

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TARGet = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TARGet

Description

This command sets/gets the the target value to be searched with the multi target search function, for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Target value for multi target search
Data Type	Double precision floating point type (Double)
Range	-5E8 to 5E8
Preset Value	0
Unit	Varies depending on the data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): ° (degree) • Group delay (GDEL): s (second) • Others: No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim TargVal as Double
SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TARGet = 12.5
TargVal = SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TARGet
```

Related Objects

Equivalent Key

Marker Search > **Multi Target** > **Target Value**

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TARGet
<numeric>
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TARGet?
```

Query Response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TARG 5"
20 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TARG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TRACking

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TRACking = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TRACking

Description

This command turns ON/OFF the search tracking (function to repeat search for each sweep) of the multi search, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the marker search tracking
Data Type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the notch search result display. • False or OFF: Turns OFF the notch search result display.
Preset Value	False or OFF

Examples

```
Dim StsTrac as Boolean
SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TRACking = True
StsTrac = SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TRACking
```

Related Objects
Equivalent Key

Marker Search > **Tracking**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TRACking {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TRACking?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TRAC ON"
20 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TRAC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TTRansition

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TTRansition = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TTRansition

Description

This command sets/gets the transition type of the multi target search, for the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Transition type of multi target search
Data Type	Character string type (String)
Range	Select from the following: • "POSitive": Specifies the positive transition. • "NEGative": Specifies the negative transition. • "BOTH": Specifies both the positive transition and the negative transition.
Preset Value	"BOTH"

Examples

```
Dim TargTran as String
SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TTRansition = "neg"
TargTran = SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TTRansition
```

Related Objects
Equivalent Key

Marker Search > **Multi Target** > **Target Transition** > **Positive**|**Negative**|**Both**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TTRansition {POSitive|NEGative|BOTH}

:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TTRansition?

Query Response

{POS|NEG|BOTH} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TTR POS"
20 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TTR?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TYPE

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MARKer.FUNCtion.MULTi.TYPE

Description

This command sets/gets the search type of the multi search, for the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Search type of the multi search
Data Type	Character string type (String)
Range	Select from the following: • "OFF": Turn OFF the multi search function. • "PEAK": Sets the search type to the multi peak search. • "TARGet": Sets the search type to the multi target search.
Preset Value	"OFF"

Examples

```
Dim SrchType as String
SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TYPE = "targ"
SrchType = SCPI.CALCulate(4).SELected.MARKer.FUNCtion.MULTi.TYPE
```

Related Objects
Equivalent Key

Marker Search > **Multi Peak** > **Search Multi Peak**

Marker Search > **Multi Target** > **Search Multi Target**

NOTE When performing the operation from the front panel, you select the search type and execute the search at the same time.

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TYPE
{OFF|PEAK|TARGet}
:CALCulate{[1]-4}[:SELected]:MARKer:FUNCtion:MULTi:TYPE?
```

Query Response

```
{OFF|PEAK|TARG} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TYPE TARG"
20 OUTPUT 717;":CALC1:MARK:FUNC:MULT:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FLATness.DATA

Object Type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FLATness.DATA

Description

This command gets the marker flatness values of the active trace, for the selected channel (*Ch)*.

Variable

Parameter	*Data*
Description	Indicates 4-elements array data (flatness value). • *Data(0)* :Span • *Data(1)* :Gain • *Data(2)* :Slope • *Data(3)* :Flatness The index of the array starts from 0.
Data Type	Variant type (Variant)

Examples

```
Dim FlatData as Variant
FlatData = SCPI.CALCulate(4).SELected.MARKer.MATH.FLATness.DATA
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.MATH.FLATness.STATe

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:FLATness:DATA?

Query Response

{numeric 1},{numeric 2},{numeric 3},{numeric 4}<newline><^END>

Parameter	**Description**

{numeric 1}	Span value
{numeric 2}	Gain value
{numeric 3}	Slope value
{numeric 4}	Flatness value

Example of use

```
10 OUTPUT 717;":CALC1:MARK:MATH:FLAT:DATA?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FLATness.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FLATness.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FLATness.STATe

Description

This command turns ON/OFF the marker flatness values display, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the marker flatness value display
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the marker flatness value display. • False or OFF: Turns OFF the marker flatness value display.
Preset Value	False or OFF

Examples

```
Dim FlatMode as Boolean
SCPI.CALCulate(4).SELected.MARKer.MATH.FLATness.STATe = True
FlatMode = SCPI.CALCulate4.SELected.MARKer.MATH.FLATness.STATe
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.MATH.FLATness.DATA

Equivalent Key

Marker Fctn > **Flatness**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:FLATness[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:FLATness[:STATe]?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK1:MATH:FLAT ON"
20 OUTPUT 717;":CALC1:MARK1:MATH:FLAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FSTatistics.DATA

Object Type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FSTatistics.DATA

Description

This command gets the RF filter statistics values of the active trace, for the selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates 3-element array data (RF filter statistics value). • *Data(0)* :Loss • *Data(1)* :Ripple (peak to peak) • *Data(2)* :Attenuation The index of the array starts from 0.
Data Type	Variant type (Variant)

Examples

```
Dim FstData as Variant
FstData = SCPI.CALCulate(4).SELected.MARKer.MATH.FSTatistics.DATA
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.MATH.FSTatistics.STATe

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:FSTatistics:DATA?

Query Response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Parameter	Description
{numeric 1}	Loss value

{numeric 2}	Ripple value (peak to peak)
{numeric 3}	Attenuation value

Example of use

```
10 OUTPUT 717;":CALC1:MARK:MATH:FST:DATA ?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FSTatistics.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FSTatistics.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.FSTatistics.STATe

Description

This command turns ON/OFF the RF filter statistics values display, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the RF filter statistics values display
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the RF filter statistics value display. • False or OFF: Turns OFF the RF filter statistics value display.
Preset Value	False or OFF

Examples

```
Dim FstMode as Boolean
SCPI.CALCulate(4).SELected.MARKer.MATH.FSTatistics.STATe = True
FstMode = SCPI.CALCulate(4).SELected.MARKer.MATH.FSTatistics.STATe
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.MATH.FSTatistics.DATA

Equivalent Key

Marker Fctn > **RF Filter Stats**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:FSTatistics[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:FSTatistics[:STATe]?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:MATH:FST ON"
20 OUTPUT 717;":CALC1:MARK:MATH:FST?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.STATistics.DATA

Object Type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.STATistics.DATA

Description

This command gets the statistics values of the active trace, for the selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates 4-element array data (statistics value). • *Data(0)* :Span value • *Data(1)* :Mean value • *Data(2)* :Standard deviation • *Data(3)* :Difference between the maximum value and the minimum value (Peak to Peak) The index of the array starts from 0.
Data Type	Variant type (Variant)

Examples

```
Dim StatData as Variant
StatData = SCPI.CALCulate(4).SELected.MARKer.MATH.STATistics.DATA
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.MATH.STATistics.STATe

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:STATistics:DATA?

Query Response

{numeric 1},{numeric 2},{numeric 3},{numeric 4}<newline><^END>

Parameter	Description

{numeric 1}	Span value
{numeric 2}	Mean value
{numeric 3}	Standard deviation value
{numeric 4}	Peak to Peak Value

Example of use

```
10 OUTPUT 717;":CALC1:MARK:MATH:STAT:DATA ?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.STATistics.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.STATistics.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.MATH.STATistics.STATe

Description

This command sets/gets turns ON/OFF the statistics values display, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the statistics values display
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the statistics value display. • False or OFF: Turns OFF the statistics value display.
Preset Value	False or OFF

Examples

```
Dim StatMode as Boolean
SCPI.CALCulate(4).SELected.MARKer.MATH.STATistics.STATe = True
StatMode = SCPI.CALCulate(4).SELected.MARKer.MATH.STATistics.STATe
```

Related Objects

SCPI.CALCulate(Ch).SELected.MARKer.MATH.STATistics.DATA

Equivalent Key

Marker Fctn > **Statistics**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:STATistics[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:MATH:STATistics[:STATe]?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:MATH:STAT ON"
20 OUTPUT 717;":CALC1:MARK:MATH:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.NOTCh.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.NOTCh.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.NOTCh.STATe

Description

This command turns ON/OFF the notch search result display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the notch search result display
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the notch search result display. • False or OFF: Turns OFF the notch search result display.
Preset value	False or OFF

Examples

```
Dim NotchSrch As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer.NOTCh.STATe = True
NotchSrch = SCPI.CALCulate(1).SELected.MARKer.NOTCh.STATe
```

Related Objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MARKer(Mk).NOTCh.DATA

SCPI.CALCulate(Ch).SELected.MARKer(Mk).NOTCh.THReshold

Equivalent key

Marker Search > **Notch**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:NOTCh[:STATe] {ON|OFF|1|0}
:CALCulate{[1]-4}[:SELected]:MARKer:NOTCh[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:NOTC ON"
20 OUTPUT 717;":CALC1:MARK:NOTC?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MARKer.REFerence.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MARKer.REFerence.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MARKer.REFerence.STATe

Description

Turns ON/OFF the reference marker mode for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the reference marker mode
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the reference marker mode. • False or OFF: Turns OFF the reference marker mode.
Preset value	False or OFF

Examples

```
Dim RefMode As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MARKer.REFerence.STATe = True
RefMode = SCPI.CALCulate(1).SELected.MARKer.REFerence.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Marker > **Ref Marker Mode**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MARKer:REFerence[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:MARKer:REFerence[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MARK:REF ON"
20 OUTPUT 717;":CALC1:MARK:REF?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.MATH.FUNCtion

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MATH.FUNCtion = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.MATH.FUNCtion

Description

This command sets/gets the data trace display method (math method between measurement data and memory trace data), for the active trace of the selected channel (*Ch*).

NOTE The math result according to this setting is displayed on the data trace.

Variable

Parameter	*Param*
Description	Math method between measurement data and memory trace data
Data type	Character string type (String)
Range	Select from the following: • "NORMal": Specifies *Data* (no math). • "DIVide": Specifies *Data / Mem*. • "MULTiply": Specifies *Data* × *Mem*. • "SUBTract": Specifies *Data - Mem*. • "ADD": Specifies *Data + Mem*. Where *Data* is the measurement data (corrected data array) and *Mem* is the data stored in the memory trace (corrected memory array).
Preset value	"NORMal"

Examples

```
Dim MathFunc As String
SCPI.CALCulate(1).PARameter(1).SELect
```

```
SCPI.CALCulate(1).SELected.MATH.FUNCtion = "div"
MathFunc = SCPI.CALCulate(1).SELected.MATH.FUNCtion
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Display > **Data Math** > **OFF** | **Data / Mem** | **Data * Mem** |**Data - Mem** | **Data + Mem**

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:MATH:FUNCtion {NORMal|
SUBTract|DIVide|ADD|MULTiply}
:CALCulate{[1]-4}[:SELected]:MATH:FUNCtion?
```

Query response

```
{NORM|DIV|MULT|SUBT|ADD}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:MATH:FUNC DIV"
20 OUTPUT 717;":CALC1:MATH:FUNC?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.MATH.MEMorize

Object type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MATH.MEMorize

Description

This command copies the measurement data to the memory trace, for the active trace of selected channel (*Ch*).

Examples

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MATH.MEMorize

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Display > **Data -> Mem**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MATH:MEMorize

Example of use

```
10 OUTPUT 717;":CALC1:MATH:MEM"
```

SCPI.CALCulate(*Ch*).SELected.MSTatistics.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.MSTatistics.DATA

Description

This command reads the statistics values of the active trace of t selected channel (*Ch*). The statistical values contain: mean value, standard deviation and the difference between the maximum value and the minimum value.

Variable

Parameter	*Data*
Description	Indicates 3-element array data (statistics value). • *Data(0)* :Mean value • *Data(1)* :Standard deviation • *Data(2)* :Difference between the maximum value and the minimum value (Peak to Peak) The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim MstData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
MstData = SCPI.CALCulate(1).SELected.MSTatistics.DATA
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MSTatistics.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MSTatistics:DATA?

Query response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Parameter	Description
{numeric 1}	Mean value
{numeric 2}	Standard deviation
{numeric 3}	Difference between the maximum value and the minimum value (Peak to Peak)

Example of use

```
10 OUTPUT 717;":CALC1:MST:DATA?"
20 ENTER 717;A,B,C
```

SCPI.CALCulate(*Ch*).SELected.MSTatistics.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.MSTatistics.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.MSTatistics.STATe

Description

This command turns ON/OFF the statistics values display, for the active trace of selected channel (*Ch*). The statistical values contain: mean value, standard deviation and the difference between the maximum value and the minimum value.

Variable

Parameter	*Status*
Description	ON/OFF of the statistics value display
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the statistics value display. • False or OFF: Turns OFF the statistics value display.
Preset value	False or OFF

Examples

```
Dim Mst As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.MSTatistics.STATe = True
Mst = SCPI.CALCulate(1).SELected.MSTatistics.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.MSTatistics.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:MSTatistics[:STATe] {ON|OFF|1|0}
:CALCulate{[1]-4}[:SELected]:MSTatistics[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:MST ON"
20 OUTPUT 717;":CALC1:MST?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.RLIMit.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.RLIMit.DATA = *Data*

Data = SCPI.CALCulate(*Ch*).SELected.RLIMit.DATA

Description

This command sets/gets the ripple limit table for the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command) of the selected channel (*Ch*).

The data transfer format when this command is executed depends on the setting with the SCPI.FORMat.DATA command.

Variable

Parameter	*Data*
Description	Indicates the array data (for ripple line) of 1 + Num (number of limit lines)×4. Where n is an integer between 1 and Num. • *Data(0)* :The number of limit lines you want to set. Specify an integer ranging from 0 to 12. When the number of limit lines is set to 0 (clears the limit table), the variable Data is only required with *Data(0)*. • *Data(n×4-3)* :The type of the n-th line. Specify an integer 0 to 1 as follows. 0: OFF 1: ON • *Data(n×4-2)* :The value on the horizontal axis (frequency/power/time) of the start point of the n-th line. • *Data(n×4-1)* :The value on the horizontal axis (frequency/power/time) of the end point of the n-th line. • *Data(n×4)* :The ripple line value (dB) of the n-th line. The index of the array starts from 0.
Data type	Variant type (Variant)

Note	If there is no array data of 1+Num (number of set lines))×4 when setting a formatted memory array, an error occurs when executed and the object is ignored. For *Data(n×4-3)* in the array data, if you specify an integer other than 0 or 1, an error occurs when executed. For *Data(n×4-2)* and *Data(n×4-1)* in the array data, if the specified value is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples (1)

```
Dim RLimData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = Array(1,1,1E6,1E9,0)
RLimData = SCPI.CALCulate(1).SELected.RLIMit.DATA

SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = Array(0) ''' Clear Ripple Limit Table
```

Examples (2)

```
Dim RlimData(5) As Variant
Dim Ref As Variant
RLimData(0) = 1
RLimData(1) = 1
RLimData(2) = 1e6
RLimData(3) = 1e9
RLimData(4) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = RLimData
Ref = SCPI.CALCulate(1).SELected.RLIMit.DATA

Dim RLimData(0) as Variant
RLimData(0) = 0
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.DATA = RLimData ''' Clear Ripple Limit Table
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

Equivalent key

Analysis > **Ripple Limit** > **Edit Ripple Limit** > **Add** |**Clear** |**Delete**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:RLIMit:DATA <numeric 1>,… ,<numeric 1+(N×4)>

:CALCulate{[1]-4}[:SELected]:RLIMit:DATA?

Query response

{numeric 1},… ,{numeric 1+(N×4)}<newline><^END>

Example of use

```
10 DIM B(1:2,1:4)
20 OUTPUT 717;":CALC1:RLIM:DATA 2,1,1E9,3E9,3,1,5E9,7E9,3"
30 OUTPUT 717;":CALC1:RLIM:DATA?"
40 ENTER 717;A,B(*)
```

```
10 OUTPUT 717;":CALC1:RLIM:DATA 0" ! Clear Ripple Limit Table
```

SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.LINE

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.LINE = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.LINE

Description

This command turns ON/OFF the ripple limit line display , for the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command) of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the ripple limit line display.
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the ripple limit line display. • False or OFF: Turns OFF the ripple limit line display.
Preset value	False or OFF

Examples

```
Dim RLimDisp As Boolean
SCPI.CALCulate(1).SELected.RLIMit.DISPlay.LINE = True
RLimDisp = SCPI.CALCulate(1).SELected.RLIMit.DISPLay.LINE
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue

Equivalent key

Analysis > **Ripple Limit** > **Ripple Limit**

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:RLIMit:DISPlay:LINE {ON|OFF|1|0}
:CALCulate{[1]-4}[:SELected]:RLIMit:DISPlay:LINE?
```

Query response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:RLIM:DISP:LINE ON"
20 OUTPUT 717;":CALC1:RLIM:DISP:LINE?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.SELect

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.SELect = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.SELect

Description

This command sets/gets the ripple limit band for ripple value display for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The ripple limit band.
Data type	Long integer type (Long)
Range	1 to 12
Preset value	1

Examples

```
Dim RBand As Long
SCPI.CALCulate(1).SELected.RLIMit.DISPlay.SELect = 2
RBand = SCPI.CALCulate(1).SELected.RLIMit.DISPlay.SELect
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue

Equivalent key

Analysis > **Ripple Limit** > **Ripple Band** > **1** to **12**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:RLIMit:DISPlay:SELect <numeric>

:CALCulate{[1]-4}[:SELected]:RLIMit:DISPlay:SELect?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:RLIM:DISP:SEL 5"
20 OUTPUT 717;":CALC1:RLIM:DISP:SEL?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.VALue

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.VALue = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.RLIMit.DISPlay.VALue

Description

This command sets/gets the display type of ripple value for the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command) of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The displaying type of ripple value.
Data type	Character string type (String)
Range	Select from the following: • "OFF": Specifies the display OFF. • "ABSolute": Specifies the absolute value for display type. • "MARgin": Specifies the margin for display type.
Preset value	"OFF"

Examples

```
Dim RDisp As String
SCPI.CALCulate(1).SELected.RLIMit.DISPlay.VALue = "ABSolute"
RDisp = SCPI.CALCulate(1).SELected.RLIMit.DISPlay.VALue
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect

Equivalent key

Analysis > Ripple Limit > Ripple Value > OFF|Absolute|Margin

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:RLIMit:DISPlay:VALue
{OFF|ABSolute|MARgin}
:CALCulate{[1]-4}[:SELected]:RLIMit:DISPlay:VALue?
```

Query response

```
{OFF|ABS|MAR}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:RLIM:DISP:VAL ABS"
20 OUTPUT 717;":CALC1:RLIM:DISP:VAL?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.RLIMit.FAIL

Object type

Property (**Read Only**)

Syntax

Status = SCPI.CALCulate(*Ch*).SELected.RLIMit.FAIL

Description

This command reads the ripple test result for the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command) of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	The ripple test result
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the ripple test result is FAIL. • False or OFF: Turns OFF the ripple test result is FAIL.
Note	When the ripple test if set to OFF, False or OFF is always read out.

Examples

```
Dim Result As Boolean
Result = SCPI.CALCulate(1).SELected.RLIMit.FAIL
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:RLIMit:FAIL?

Example of use

```
10 OUTPUT 717;":CALC1:RLIM:FAIL?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.RLIMit.REPort.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.CALCulate(*Ch*).SELected.RLIMit.REPort.DATA

Description

This command reads the ripple value of the ripple test for the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command) of the selected channel (*Ch*).

The data transfer format when this command is executed depends on the setting with the SCPI.FORMat.DATA command.(Read only)

Variable

Parameter	*Data*
Description	Indicates the array data (for ripple line) of 1 + Num (number of limit lines)×3. Where n is an integer between 1 and 12. • *Data(0)* :Number of ripple limit line. • *Data(n×3-2)* :Number of ripple limit bands. • *Data(n×3-1)* :Ripple value. • *Data(n×3)* :Results of ripple test. Select from the following. 0:PASS 1:FAIL. The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim RData As Variant
SCPI.CALCulate(1).PARameter(1).SELect
RData = SCPI.CALCulate(1).SELected.RLIMit.REPort.DATA
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:RLIMit:REPort[:DATA]?

Query response

{numeric 1},… ,{numeric 1+N×3}<newline><^END>

Type	Description
<numeric 1>	Number of ripple limit line (1 to 12)
<numeric 1+(n×3)-2>	Number of ripple limit bands
<numeric 1+(n×3)-1>	Ripple value
<numeric 1+(n×3)>	Results of ripple test 0: Pass 1: Fail

Where N is the number of lines (specified with <numeric 1>) and n is an integer between 1 and 12.

Example of use

```
10 DIM B(1:2,1:3)
20 OUTPUT 717;":CALC1:RLIM:REP?"
30 ENTER 717;A,B(*)
```

SCPI.CALCulate(*Ch*).SELected.RLIMit.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.RLIMit.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.RLIMit.STATe

Description

This command turns ON/OFF the ripple test function for the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command) of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the ripple test function
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the ripple test function. • False or OFF: Turns OFF the ripple test function.
Preset value	False or OFF

Examples

```
Dim RLimTest As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.RLIMit.STATe = True
RLimTest = SCPI.CALCulate(1).SELected.RLIMit.STATe
```

Related Objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.RLIMit.DATA

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.LINE

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.SELect

SCPI.CALCulate(Ch).SELected.RLIMit.DISPlay.VALue

SCPI.CALCulate(Ch).SELected.RLIMit.FAIL

SCPI.CALCulate(Ch).SELected.RLIMit.REPort.DATA

Equivalent key

Analysis > **Ripple Limit** > **Ripple Limit Test**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:RLIMit[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:RLIMit[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:RLIM ON"
20 OUTPUT 717;":CALC1:RLIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.SMOothing.APERture

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.SMOothing.APERture = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.SMOothing.APERture

Description

This command sets/gets the smoothing aperture (percentage to the sweep span value) of the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Smoothing aperture
Data type	Double precision floating point type (Double)
Range	0.05 to 25
Preset value	1.5
Unit	% (percent)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim SmoAper As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.SMOothing.APERture = 2.5
SmoAper = SCPI.CALCulate(1).SELected.SMOothing.APERture
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.SMOothing.STATe

Equivalent key

Avg > **Smo Aperture**

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:SMOothing:APERture <numeric>
:CALCulate{[1]-4}[:SELected]:SMOothing:APERture?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:SMO:APER 2.5"
20 OUTPUT 717;":CALC1:SMO:APER?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.SMOothing.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.SMOothing.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.SMOothing.STATe

Description

This command turns ON/OFF the smoothing, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the smoothing
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the smoothing. • False or OFF: Turns OFF the smoothing.
Preset value	False or OFF

Examples

```
Dim Smo As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.SMOothing.STATe = True
Smo = SCPI.CALCulate(1).SELected.SMOothing.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.SMOothing.APERture

Equivalent key

Avg > **Smoothing**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:SMOothing[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:SMOothing[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:SMO ON"
20 OUTPUT 717;":CALC1:SMO?"
30 ENTER 717;A
```

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.CENTer

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.CENTer = *Value*

Value = SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.CENTer

Description

This command sets/gets the center value used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Center value
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	0
Unit	ft (feet) or m (meters)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Cent as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.CENTer = 1E-8
Cent = SCPI.CALCulate(4).SELected.TRANsform.DISTance.CENTer
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STARt

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STOP

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.UNIT

Equivalent Key

Analysis > Fault Location > Center

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:CENTer <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:CENTer?
```

Query Response

```
{numeric} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:CENT 0"
20 OUTPUT 717;":CALC1:TRAN:DIST:CENT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.CLOSs

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.CLOSs = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.CLOSs

Description

This command sets/gets the cable loss value used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Cable Loss value
Data Type	Double precision floating point type (Double)
Range	Varies depending on the distance unit.
Preset Value	0
Unit	dB/100m or dB/100ft
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Closs as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.CLOSs = 10
Closs = SCPI.CALCulate(4).SELected.TRANsform.DISTance.CLOSs
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.UNIT

Equivalent Key

Analysis > **Fault Location** > **Cable Loss**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:CLOSs <numeric>

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:CLOSs?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:CLOS 0"
20 OUTPUT 717;":CALC1:TRAN:DIST:CLOS?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.IMPulse.WIDTh

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.IMPulse.WIDTh = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.IMPulse.WIDTh

Description

This command sets/gets the shape of the Kayser Bessel window using the impulse width used for the transformation function of the fault location display, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Impulse width
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and transformation type.
Preset Value	650.395679856p sec
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim ImpWid as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.IMPulse.WIDTh = 1E-10
ImpWid = SCPI.CALCulate(4).SELected.TRANsform.DISTance.IMPulse.WIDTh
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.KBESsel

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STEP.RTIMe

Equivalent Key

Analysis > Fault Location > Window > Impulse Width

Equivalent SCPI Command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:IMPulse:WIDTh <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:IMPulse:WIDTh?
```

Query Response

```
{numeric} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:IMP:WIDT 7E-10"
20 OUTPUT 717;":CALC1:TRAN:DIST:IMP:WIDT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.KBESsel

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.KBESsel = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.KBESsel

Description

This command sets/gets the shape of the Kayser Bessel window using β used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The value of β
Data Type	Double precision floating point type (Double)
Range	0 to 13
Preset Value	6
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim KBeta as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.KBESsel = 3
KBeta = SCPI.CALCulate(4).SELected.TRANsform.DISTance.KBESsel
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.IMPulse.WIDTh

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STEP.RTIMe

Equivalent Key

Analysis > **Fault Location** > **Window** > **Kaiser Beta**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:KBESsel <numeric>

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:KBESsel?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:KBES 0"
20 OUTPUT 717;":CALC1:TRAN:DIST:KBES?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.LPFRequency

Object Type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.LPFRequency

Description

This command changes the frequency range to match with the low-pass type transformation of the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Examples

SCPI.CALCulate(4).SELected.TRANsform.DISTance.LPFRequency

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STATe

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.TYPE

Equivalent Key

Analysis > **Fault Location** > **Set Freq Low Pass**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:LPFRequency

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:LPFR"
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.REFLection.TYPE

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.REFLection.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.REFLection.TYPE

Description

This command sets/gets the reflection distance either one way or round trip, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The reflection distance either one way or round trip
Data Type	Character string type (String)
Range	Select from the following: • "OWAY": Specifies the One Way. • "RTRip": Specifies the Round Trip.
Preset Value	"RTRip"

Examples

```
Dim StimType as String
SCPI.CALCulate(4).SELected.TRANsform.DISTance.REFLection.TYPE = "OWAY"
StimType = SCPI.CALCulate(4).SELected.TRANsform.DISTance.REFLection.TYPE
```

Related Objects
Equivalent Key

Analysis > **Fault Location** > **Reflection Type**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:REFLection:TYPE {OWAY|RTRip}

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:REFLection:TYPE?

Query Response

{OWAY|RTR} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:REFL:TYPE RTR"
20 OUTPUT 717;":CALC1:TRAN:DIST:REFL:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.SPAN

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.SPAN = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.SPAN

Description

This command sets/gets the span value used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Span value
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	19.6714211286 ft or 5.99584916 m
Unit	ft (feet) or m (meters)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Span as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.SPAN = 0.42E+1
Span = SCPI.CALCulate(4).SELected.TRANsform.DISTance.SPAN
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STARt

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STOP

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.UNIT

Equivalent Key

Analysis > Fault Location > Span

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:SPAN <numeric>

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:SPAN?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:SPAN 4.42"
20 OUTPUT 717;":CALC1:TRAN:DIST:SPAN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STARt

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STARt

Description

This command sets/gets the start value used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Start value
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	-9.8357105643 ft or -2.99792458 m
Unit	ft (feet) or m (meters)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Star as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.STARt = 0
Star = SCPI.CALCulate(4).SELected.TRANsform.DISTance.STARt
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STOP

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.UNIT

Equivalent Key

Analysis > Fault Location > Start

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STARt <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STARt?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:STAR 0"
20 OUTPUT 717;":CALC1:TRAN:DIST:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STATe = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STATe

Description

This command turns ON/OFF the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

You can enable the transformation function only when the sweep type is the linear sweep and the number of points is three or more. If you execute this object to try to enable the transformation function when the sweep type is other than the linear sweep or the number of points is less than three, an error occurs and the object is ignored.

NOTE When the sweep type is the power sweep, you cannot turn ON the transformation function. If you execute this object trying to turn ON the transformation function during the power sweep, an error occurs and the object is ignored.

Variable

Parameter	*Status*
Description	ON/OFF of the transformation function
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the transformation function. • False or OFF: Turns OFF the transformation function.
Preset value	False or OFF

Examples

```
Dim Trans as Boolean
SCPI.CALCulate(4).SELected.TRANsform.DISTance.STATe = True
Trans = SCPI.CALCulate(4).SELected.TRANsform.DISTance.STATe
```

Related Objects

Equivalent Key

Analysis > **Fault Location** > **Fault Location**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STATe {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STATe?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:STAT ON"
20 OUTPUT 717;":CALC1:TRAN:DIST:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STEP.RTIMe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STEP.RTIMe = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STEP.RTIMe

Description

This command sets/gets the shape of the Kayser Bessel window using the rise time of step signal used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The rise time of step signal
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	328.677622587p sec
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Rtime as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.STEP.RTIMe = 4.2E-10
Rtime = SCPI.CALCulate(4).SELected.TRANsform.DISTance.STEP.RTIMe
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.IMPulse.WIDTh

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.KBESsel

Equivalent Key

Analysis > **Fault Location** > **Window** > **Step Rise**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STEP:RTIMe <numeric>

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STEP:RTIMe?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:STEP:RTIM 4.2E-10"
20 OUTPUT 717;":CALC1:TRAN:DIST:STEP:RTIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STIMulus

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STIMulus = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STIMulus

Description

This command sets/gets the stimulus type used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The stimulus type
Data Type	Character string type (String)
Range	Select from the following: • "IMPulse": Specifies the impulse • "STEP": Specifies the step
Preset Value	"IMPulse"

Examples

```
Dim StimType as String
SCPI.CALCulate(4).SELected.TRANsform.DISTance.STIMulus = "IMPulse"
StimType = SCPI.CALCulate(4).SELected.TRANsform.DISTance.STIMulus
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STATe

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.TYPE

Equivalent Key

Analysis > **Fault Location** > **Type** > **Bandpass**|**Lowpass Step**|**Lowpass Imp**

NOTE When performing this operation from the front panel, you select the transformation type at the same time.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STIMulus {IMPulse|STEP}

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STIMulus?

Query Response

{IMP|STEP} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST LPAS"
20 OUTPUT 717;":CALC1:TRAN:DIST:STIM STEP"
30 OUTPUT 717;":CALC1:TRAN:DIST:STIM?"
40 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STOP

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.STOP

Description

This command sets/gets the stop value used for the transformation function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Stop value
Data Type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset Value	9.8357105643 ft or 2.99792458 m
Unit	ft (feet) or m (meters)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim DistStop as Double
SCPI.CALCulate(4).SELected.TRANsform.DISTance.STOP = -2.5
DistStop = SCPI.CALCulate(4).SELected.TRANsform.DISTance.STOP
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STARt

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.UNIT

Equivalent Key

Analysis > Fault Location > Stop

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STOP <numeric>

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:STOP?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:STOP -2.2"
20 OUTPUT 717;":CALC1:TRAN:DIST:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.TYPE

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.TYPE

Description

This command sets/gets the transformation type used for the transformation function of the fault location, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The transformation type
Data type	Character string type (String)
Range	Select from the following: • "BPASs": Specifies the band-pass • "LPASs": Specifies the low-pass
Preset value	"BPASs"

Examples

```
Dim Typ as String
SCPI.CALCulate(4).SELected.TRANsform.DISTance.TYPE = "LPASs"
Typ = SCPI.CALCulate(4).SELected.TRANsform.DISTance.TYPE
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STATe

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STIMulus

Equivalent Key

Analysis > **Fault Location** > **Type** > **Bandpass**|**Lowpass Step**|**Lowpass Imp**

NOTE When performing this operation from the front panel, you select the stimulus type at the same time.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance[:TYPE] {BPASs|LPASs}

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance[:TYPE]?

Query Response

{BPAS|LPAS} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST LPAS"
20 OUTPUT 717;":CALC1:TRAN:DIST?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.UNIT

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.UNIT = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.DISTance.UNIT

Description

This command sets/gets the distance unit used for the transform function of the fault location display, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The distance unit used for the transform function
Data Type	Character string type (String)
Range	Select from the following. • "METers": Specifies the meter. • "FEET": Specifies the feet.
Preset Value	"METers"

Examples

```
Dim DistUnit as String
SCPI.CALCulate(4).SELected.TRANsform.DISTance.UNIT = "METers"
DistUnit = SCPI.CALCulate(4).SELected.TRANsform.DISTance.UNIT
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.CENTer

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.SPAN

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STARt

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.STOP

Equivalent Key

Analysis > **Fault Location** > **Unit** > **Meters**|**Feet**|**Seconds**

NOTE When performing this operation from the front panel, you select the horizontal axis of the transform function either time or distance at the same time.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:UNIT {METers|FEET}

:CALCulate{[1]-4}[:SELected]:TRANsform:DISTance:UNIT?

Query Response

{MET|FEET} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:DIST:UNIT FEET"
20 OUTPUT 717;":CALC1:TRAN:DIST:UNIT?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.METHod

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.METHod = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.METHod

Description

This command sets/gets the horizontal axis of the transform function either as time or distance, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The horizontal axis of the transform function
Data Type	Character string type (String)
Range	Select from the following. • "TIME": Specifies the time. • "DISTance": Specifies the distance.
Preset Value	"TIME"

Examples

```
Dim TranMeth as String
SCPI.CALCulate(4).SELected.TRANsform.METHod = "TIME"
TranMeth = SCPI.CALCulate(4).SELected.TRANsform.METHod
```

Related Objects

SCPI.CALCulate(Ch).SELected.TRANsform.DISTance.UNIT

Equivalent Key

Analysis > **Fault Location** > **Unit** > **Meters**|**Feet**|**Seconds**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:METHod {TIME|DISTance}

:CALCulate{[1]-4}[:SELected]:TRANsform:METHod?

Query Response

{TIME|DIST} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:METH DIST"
20 OUTPUT 717;":CALC1:TRAN:METH?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CENTer

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CENTer = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CENTer

Description

This command sets/gets the center value used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Center value
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	0
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Cent As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.CENTer = 1E-8
Cent = SCPI.CALCulate(1).SELected.TRANsform.TIME.CENTer
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.SPAN

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Fault Location > Center

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:CENTer <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:CENTer?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:CENT 1E-8"
20 OUTPUT 717;":CALC1:TRAN:TIME:CENT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CLOSs

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CLOSs = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.CLOSs

Description

This command sets/gets the cable loss value used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Cable loss value
Data Type	Double precision floating point type (Double)
Range	0 to 1000
Preset Value	0
Unit	dB/µsec
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Closs as Double
SCPI.CALCulate(4).SELected.TRANsform.TIME.CLOSs = 3.4
Closs = SCPI.CALCulate(4).SELected.TRANsform.TIME.CLOSs
```

Related Objects

Equivalent Key

Analysis > **Fault Location** > **Cable Loss**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:CLOSs <numeric>

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:CLOSs?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:CLOS 0"
20 OUTPUT 717;":CALC1:TRAN:TIME:CLOS?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.IMPulse.WIDTh

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.IMPulse.WIDTh = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.IMPulse.WIDTh

Description

This command sets/gets the shape of the Kayser Bessel window using the impulse width used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Impulse width
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and transformation type.
Preset value	Varies depending on the frequency span and transformation type.
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim ImpWid As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.IMPulse.WIDTh = 1E-10
ImpWid = SCPI.CALCulate(1).SELected.TRANsform.TIME.IMPulse.WIDTh
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME. KBESsel

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STEP.RTIMe

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > **Fault Location** > **Window** > **Impulse Width**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:IMPulse:WIDTh <numeric>

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:IMPulse:WIDTh?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:IMP:WIDT 1E-10"
20 OUTPUT 717;":CALC1:TRAN:TIME:IMP:WIDT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.KBESsel

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.KBESsel = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.KBESsel

Description

This command sets/gets the shape of the Kayser Bessel window used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The value of b
Data type	Double precision floating point type (Double)
Range	0 to 13
Preset value	6
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Beta As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.KBESsel = 3
Beta = SCPI.CALCulate(1).SELected.TRANsform.TIME.KBESsel
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME. IMPulse.WIDTh

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STEP.RTIMe

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Fault Location > Window > Kaiser Beta

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:KBESsel <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:KBESsel?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:KBES 3"
20 OUTPUT 717;":CALC1:TRAN:TIME:KBES?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.LPFRequency

Object type

Property (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.LPFRequency = *Value*

Description

This command changes the frequency range to match with the low-pass type transformation of the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Examples

```
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.LPFRequency = 1E5
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.TYPE

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > **Fault Location** > **Set Freq Low pass**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:LPFRequency

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:LPFR 1E5"
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.REFLection.TYPE

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.REFLection.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.REFLection.TYPE

Description

This command sets/gets the reflection time either one way or round trip, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The reflection time either one way or round trip
Data Type	Character string type (String)
Range	Select from the following: • "OWAY": Specifies the One Way. • "RTRip": Specifies the Round Trip.
Preset Value	"RTRip"

Examples

```
Dim RefType as String
SCPI.CALCulate(4).SELected.TRANsform.TIME.REFLection.TYPE = "RTRip"
RefType = SCPI.CALCulate(4).SELected.TRANsform.TIME.REFLection.TYPE
```

Related Objects

Equivalent Key

Analysis > **Fault Location** > **Reflection Type**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:REFLection:TYPE {OWAY|RTRip}

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:REFLection:TYPE?

Query Response

{OWAY|RTR} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:REFL:TYPE OWAY"
20 OUTPUT 717;":CALC1:TRAN:TIME:REFL:TYPE?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.SPAN

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.SPAN = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.SPAN

Description

This command selects the span value used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Span value
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	2E-8
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Span As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.SPAN = 1E-8
Cent = SCPI.CALCulate(1).SELected.TRANsform.TIME.SPAN
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.CENTer

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Fault Location > Center

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:SPAN <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:SPAN?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:SPAN 1E-8"
20 OUTPUT 717;":CALC1:TRAN:TIME:SPAN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STARt

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STARt = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STARt

Description

This command sets/gets the start value used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Start value
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	-1E-8
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Star As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STARt = 0
Star = SCPI.CALCulate(1).SELected.TRANsform.TIME.STARt
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STOP

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Fault Location > Start

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STARt <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STARt?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:STAR 0"
20 OUTPUT 717;":CALC1:TRAN:TIME:STAR?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STATe

Description

This command turns ON/OFF the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

You can enable the transformation function only when the sweep type is the linear sweep and the number of points is three or more. If you execute this object to enable the transformation function when the sweep type is other than the linear sweep or the number of points is less than three, an error occurs and the object is ignored.

NOTE When the sweep type is the power sweep, you cannot turn ON the transformation function. If you execute this object to turn ON the transformation function during the power sweep, an error occurs and the object is ignored.

Variable

Parameter	*Status*
Description	ON/OFF of the gating function
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the transformation function. • False or OFF: Turns OFF the transformation function.
Preset value	False or OFF

Examples

```
Dim Trans As Boolean
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STATe = True
Trans = SCPI.CALCulate(1).SELected.TRANsform.TIME.STATe
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SENSe(Ch).SWEep.POINts

Equivalent key

Analysis > **Fault Location** > **Transform**

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STATe {ON|OFF|1|0}

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:STAT ON"
20 OUTPUT 717;":CALC1:TRAN:TIME:STAT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STEP.RTIMe

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STEP.RTIMe = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STEP.RTIMe

Description

This command sets/gets the shape of the Kayser Bessel window using the rise time of step signal used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The rise time of step signal
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span.
Preset value	Varies depending on the frequency span.
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim RTime As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STEP.RTIMe = 1E-10
RTime = SCPI.CALCulate(1).SELected.TRANsform.TIME.STEP.RTIMe
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.IMPulse.WIDTh

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.KBESsel

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > **Fault Location** > **Center**

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STEP:RTIMe <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STEP:RTIMe?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:STEP:RTIM 1E-10"
20 OUTPUT 717;":CALC1:TRAN:TIME:STEP:RTIM?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STIMulus

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STIMulus = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STIMulus

Description

This command sets/gets the stimulus type used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The stimulus type
Data type	Character string type (String)
Range	Select from the following: • "IMPulse": Specifies the impulse • "STEP": Specifies the step
Preset value	"IMPulse"

Examples

```
Dim StimType As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STIMulus = "step"
StimType = SCPI.CALCulate(1).SELected.TRANsform.TIME.STIMulus
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.TYPE

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > **Fault Location** > **Type** > **Bandpass**|**Lowpass Step**|**Lowpass Imp.**

NOTE When performing this operation from the front panel, you select the transformation type at the same time.

Equivalent SCPI command

Syntax

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STIMulus {IMPulse|STEP}

:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STIMulus?

Query response

{IMP|STEP}<newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME LPAS"
20 OUTPUT 717;":CALC1:TRAN:TIME:STIM STEP"
30 OUTPUT 717;":CALC1:TRAN:TIME:STIM?"
40 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STOP

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STOP = *Value*

Value = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.STOP

Description

This command sets/gets the stop value used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Stop value
Data type	Double precision floating point type (Double)
Range	Varies depending on the frequency span and the number of points.
Preset value	1E-8
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Span As Double
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.STOP = 2E-8
Cent = SCPI.CALCulate(1).SELected.TRANsform.TIME.STOP
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STARt

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > Fault Location > Stop

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STOP <numeric>
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME:STOP?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME:STOP 2E-8"
20 OUTPUT 717;":CALC1:TRAN:TIME:STOP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.TYPE

Object type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.TYPE = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.TRANsform.TIME.TYPE

Description

This command sets/gets the transformation type used for the transformation function of the time domain function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The transformation type
Data type	Character string type (String)
Range	Select from the following: • "BPASs": Specifies the band-pass • "LPASs": Specifies the low-pass
Preset value	"BPASs"

Examples

```
Dim Typ As String
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.CALCulate(1).SELected.TRANsform.TIME.SHAPe = "lpas"
Typ = SCPI.CALCulate(1).SELected.TRANsform.TIME.SHAPe
```

Related objects

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STIMulus

SCPI.CALCulate(Ch).SELected.TRANsform.TIME.STATe

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Analysis > **Fault Location** > **Type** > **Bandpass**|**Lowpass Step**|**Lowpass Imp.**

When performing this operation from the front panel, you select the stimulus type at the same time.

Equivalent SCPI command

Syntax

```
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME[:TYPE] {BPASs|LPASs}
:CALCulate{[1]-4}[:SELected]:TRANsform:TIME[:TYPE]?
```

Query response

```
{BPAS|LPAS}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CALC1:TRAN:TIME LPAS"
20 OUTPUT 717;":CALC1:TRAN:TIME?"
30 ENTER 717;A$
```

SCPI.CALCulate(*Ch*).SELected.ZPARameter.DEFine

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SELected.ZPARameter.DEFine = *Param*

Param = SCPI.CALCulate(*Ch*).SELected.ZPARameter.DEFine

Description

This command sets/gets the measurement parameter of the selected trace (*Tr*), for the selected channel (*Ch*). When you define the parameters, follows the sequence shown in the examples below.

Variable

Parameter	*Param*
Description	Measurement parameter
Data Type	Character string type (String)
Range	Select one of the following parameters: Z|Y|Cp|Cs|Lp|Ls|Rp|Rs|D|Q
Preset Value	Z
Unit	-
Resolution	-

Examples

S-Parameter measurement port

For $|Z|$,$|Y|$

```
SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z" (or "Y")
SCPI.CALCulate(1).SELected.FORMat = "MLINear"
```

For Cp, Cs, Lp, Ls, Rp, Rs, D, Q

```
SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
```

SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "CP" (or "CS", "LP", "LS", "RP, "RS" ,"D", "Q")
SCPI.CALCulate(1).SELected.FORMat = "REAL"

For θZ

SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.FORMat = "Phase"

For |Y|

SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "MLINear"

For θy

SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "Phase"

For R

SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.FORMat = "Real"

For X

SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.FORMat = "Imag"

For G

SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "Real"

For B

SCPI.CALCulate(1).PARameter.DEFine = "S11"
SCPI.SENSe.Z.METHod = "P1Reflection" (or "P2Reflection", "TSERies", "TSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "Imag"

Gain-Phase measurement port

For |Z|,|Y|

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z" (or "Y")
SCPI.CALCulate(1).SELected.FORMat = "MLINear"

For Cp, Cs, Lp, Ls, Rp, Rs, D, Q

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "CP" (or "CS", "LP", "LS", "RP, "RS" ,"D", "Q")
SCPI.CALCulate(1).SELected.FORMat = "REAL"

For θZ

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.FORMat = "Phase"

For |Y|

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "MLINear"

For θy

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "Phase"

For R

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.FORMat = "Real"

For X

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"

SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.FORMat = "Imag"

For G

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "Real"

For B

SCPI.CALCulate(1).PARameter.DEFine = "TR"
SCPI.SENSe.Z.METHod = "GSERies" (or "GSHunt")
SCPI.CALCulate(1).PARameter.DEFine = "Z"
SCPI.CALCulate(1).SELected.ZPARameter.DEFine = "Y"
SCPI.CALCulate(1).SELected.FORMat = "Imag"

Related Objects

SCPI.CALCulate(Ch).PARameter(Tr).DEFine

SCPI.CALCulate(Ch).SELected.FORMat

Equivalent Key

Meas > **Impedance Analysis Menu** > **|Z|** or

Meas > **Impedance Analysis Menu** > **|Y|** or

Meas > **Impedance Analysis Menu** > **Cp** or

Meas > **Impedance Analysis Menu** > **Cs** or

Meas > **Impedance Analysis Menu** > **Lp** or

Meas > **Impedance Analysis Menu** > **Ls** or

Meas > **Impedance Analysis Menu** > **Rp** or

Meas > **Impedance Analysis Menu** > **Rs** or

Meas > **Impedance Analysis Menu** > **D** or

Meas > **Impedance Analysis Menu** > **Q**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}[:SELected]:ZPARameter:DEFine <string>

:CALCulate{[1]-4}[:SELected]:ZPARameter:DEFine?

Query Response

<string><newline><^END>

Example of use

See the examples in COM

SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).CAPacitance

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).CAPacitance = *Value*

Value = SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).CAPacitance

Description

This command sets/gets the connector capacitance value of the specified port (*Pt*) for the connector mismatch compensation, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Connector capacitance for compensation
Data Type	Double precision floating point type (Double)
Range	-2E-12 to 2E-12
Preset Value	0
Unit	F (farad)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim ConnCap as Double
SCPI.CALCulate(4).SRL.CONNector(2).CAPacitance = 1E-3
ConnCap = SCPI.CALCulate(4).SRL.CONNector(2).CAPacitance
```

Related Objects

SCPI.CALCulate(Ch).SRL.CONNector(Pt).IMMediate

SCPI.CALCulate(Ch).SRL.CONNector(Pt).LENGth

Equivalent Key

Analysis > **SRL** > **Port1 Connector**|**Port2 Connector** > **Capacitance**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL:CONNector{[1]|2}:CAPacitance <numeric>

:CALCulate{[1]-4}:SRL:CONNector{[1]|2}:CAPacitance?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:SRL:CONN2:CAP 1.2E-2"
20 OUTPUT 717;":CALC1:SRL:CONN2:CAP?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).IMMediate

Object Type

Method (**Write Only**)

Syntax

SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).IMMediate

Description

This command measures the terminated cable connected to the specified port (*Pt*) and automatically sets the connector length and capacitance values for connector mismatch compensation, for the active trace of the selected channel (*Ch*).

Examples

SCPI.CALCulate(2).SRL.CONNector(1).IMMediate

Related Objects

SCPI.CALCulate(Ch).SRL.CONNector(Pt).CAPacitance

SCPI.CALCulate(Ch).SRL.CONNector(Pt).LENGth

Equivalent Key

Analysis > **SRL** > **Port1 Connector|Port2 Connector** > **Measure Connector**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL:CONNector{[1]|2}:IMMediate

Example of use

```
10 OUTPUT 717;":CALC1:SRL:CONN2:IMM"
```

SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).IMPedance

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).IMPedance

Description

This command gets the average cable impedance of specified port (*Pt*) for the SRL calculation, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Impedance value
Data Type	Double precision floating point type (Double)
Unit	Ω

Examples

```
Dim PortImp as Double
PortImp = SCPI.CALCulate(2).SRL.CONNector(2).IMPedance
```

Related Objects

SCPI.CALCulate(Ch).SRL.CONNector(Pt).IMMediate

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL:CONNector{[1]|2}:IMPedance?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:SRL:CONN2:IMP?"
20 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).LENGth

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).LENGth = *Value*

Value = SCPI.CALCulate(*Ch*).SRL.CONNector(*Pt*).LENGth

Description

This command sets/gets the connector length value of specified port (*Pt*) for the connector mismatch compensation, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Connector length value for compensation
Data Type	Double precision floating point type (Double)
Range	-0.02 to 0.2
Preset Value	0
Unit	m (meters)

Examples

```
Dim Leng as Double
SCPI.CALCulate(2).SRL.CONNector(1).LENGth = -0.01
Leng = SCPI.CALCulate(2).SRL.CONNector(1).LENGth
```

Related Objects

SCPI.CALCulate(Ch).SRL.CONNector(Pt).CAPacitance

SCPI.CALCulate(Ch).SRL.CONNector(Pt).IMMediate

Equivalent Key

Analysis > **SRL** > **Port1 Connector**|**Port2 Connector** > **Length**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL:CONNector{[1]|2}:LENGth <numeric>

:CALCulate{[1]-4}:SRL:CONNector{[1]|2}:LENGth?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:SRL:CONN2:LENG -0.02"
20 OUTPUT 717;":CALC1:SRL:CONN2:LENG?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SRL.IMPedance.AUTO.CUToff

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SRL.IMPedance.AUTO.CUToff = *Value*

Value = SCPI.CALCulate(*Ch*).SRL.IMPedance.AUTO.CUToff

Description

This command sets/gets the cutoff frequency of the auto calculation for the average cable impedance, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The cutoff frequency of the auto calculation
Data Type	Double precision floating point type (Double)
Range	5 to 3E+9
Preset Value	2.1E+8
Unit	Hz
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim CutFreq as Double
SCPI.CALCulate(2).SRL.IMPedance.AUTO.CUToff = 2.1E9
CutFreq = SCPI.CALCulate(2).SRL.IMPedance.AUTO.CUToff
```

Related Objects

SCPI.CALCulate(Ch).SRL.IMPedance.AUTO.STATe

SCPI.CALCulate(Ch).SRL.IMPedance.MANual

Equivalent Key

Analysis > **SRL** > **Z Cutoff Freq**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL:IMPedance:AUTO:CUToff <numeric>
:CALCulate{[1]-4}:SRL:IMPedance:AUTO:CUToff?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:SRL:IMP:AUTO:CUT 0"
20 OUTPUT 717;":CALC1:SRL:IMP:AUTO:CUT?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SRL.IMPedance.AUTO.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SRL.IMPedance.AUTO.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SRL.IMPedance.AUTO.STATe

Description

This command turns ON/OFF the auto impedance calculation function of the SRL measurement, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Auto impedance calculation status
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the auto impedance calculation function. • False or OFF: Turns OFF the auto impedance calculation function.
Preset Value	True or ON

Examples

```
Dim Stat as Boolean
SCPI.CALCulate(4).SRL.IMPedance.AUTO.STATe = True
Stat = SCPI.CALCulate(4).SRL.IMPedance.AUTO.STATe
```

Related Objects

SCPI.CALCulate(Ch).SRL.IMPedance.AUTO.CUToff

SCPI.CALCulate(Ch).SRL.IMPedance.MANual

Equivalent Key

Analysis > **SRL** > **Auto Z**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL:IMPedance:AUTO[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}:SRL:IMPedance:AUTO[:STATe]?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC2:SRL:IMP:AUTO ON"
20 OUTPUT 717;":CALC2:SRL:IMP:AUTO?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SRL.IMPedance.MANual

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SRL.IMPedance.MANual = *Value*

Value = SCPI.CALCulate(*Ch*).SRL.IMPedance.MANual

Description

This command sets/gets the average cable impedance to be used in the SRL calculation, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The average cable impedance value
Data Type	Double precision floating point type (Double)
Range	10 to 1000
Preset Value	50
Unit	Ω

Examples

```
Dim Imp as Double
SCPI.CALCulate(1).SRL.IMPedance.MANual = 75
Imp = SCPI.CALCulate(1).SRL.IMPedance.MANual
```

Related Objects

SCPI.CALCulate(Ch).SRL.IMPedance.AUTO.CUToff

SCPI.CALCulate(Ch).SRL.IMPedance.AUTO.STATe

Equivalent Key

Analysis > **SRL** > **Manual Z**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL:IMPedance:MANual <numeric>

:CALCulate{[1]-4}:SRL:IMPedance:MANual?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:SRL:IMP:MAN 0"
20 OUTPUT 717;":CALC1:SRL:IMP:MAN?"
30 ENTER 717;A
```

SCPI.CALCulate(*Ch*).SRL.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.CALCulate(*Ch*).SRL.STATe = *Status*

Status = SCPI.CALCulate(*Ch*).SRL.STATe

Description

This command turns ON/OFF the SRL measurement function, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	SRL measurement function status
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the SRL measurement function. • False or OFF: Turns OFF the SRL measurement function.
Preset Value	False or OFF

Examples

```
Dim SRLStat as Boolean
SCPI.CALCulate(2).SRL.STATe = True
SRLStat = SCPI.CALCulate(2).SRL.STATe
```

Related Objects

SCPI.CALCulate(Ch).SRL.CONNector(Pt).CAPacitance

SCPI.CALCulate(Ch).SRL.CONNector(Pt).LENGth

SCPI.CALCulate(Ch).SRL.IMPedance.MANual

Equivalent Key

Analysis > **SRL** > **SRL**

Equivalent SCPI Command

Syntax

:CALCulate{[1]-4}:SRL[:STATe] {ON|OFF|1|0}

:CALCulate{[1]-4}:SRL[:STATe]?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":CALC1:SRL ON"
20 OUTPUT 717;":CALC1:SRL?"
30 ENTER 717;A
```

CONTROL

SCPI.CONTrol.HANDler.A.DATA

Object type

Method (**Write Only**)

Syntax

SCPI.CONTrol.HANDler.A.DATA = *Value*

Description

This command sets/gets information of output port A (A0 to A7) of the handler I/O. Port information is output as 8-bit binary data using A0 as LSB and A7 as MSB.

Variable

Parameter	*Value*
Description	Port information (output)
Data type	Long integer type (Long)
Range	0 to 255
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Example of use

SCPI.CONTrol.HANDler.A.DATA = 15

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler:A[:DATA] <numeric>

Example of use

```
10 OUTPUT 717;":CONT:HAND:A 15"
20 OUTPUT 717;":CONT:HAND:A:DATA 15"
```

SCPI.CONTrol.HANDler.B.DATA

Object type

Method (**Write Only**)

Syntax

SCPI.CONTrol.HANDler.B.DATA = *Value*

Description

This command sets/gets information of output port B (B0 to B7) of the handler I/O. Port information is output as 8-bit binary data using B0 as LSB and B7 as MSB.

NOTE The bit 6 of the data outputted by this project is ignored when outputting the INDEX signal is turned ON (specifying True with the SCPI.CONTrol.HANDler.EXTension.INDex.STATe object).

NOTE The bit 7 of the data outputted by this project is ignored when outputting the READY FOR TRIGGER signal is turned ON (specifying True with the SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe object).

Variable

Parameter	*Value*
Description	Port information (output)
Data type	Long integer type (Long)
Range	0 to 255
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
SCPI.CONTrol.HANDler.B.DATA = 15
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler:B[:DATA] <numeric>

Example of use

```
10 OUTPUT 717;":CONT:HAND:B 15"
20 OUTPUT 717;":CONT:HAND:B:DATA 15"
```

SCPI.CONTrol.HANDler.C.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.C.DATA = *Value* (for output port)

Value = SCPI.CONTrol.HANDler.C.DATA (for input port)

Description

When input/output port C of the handler I/O is set to the output port, it outputs port information to the output port C (C0 to C3).

When input/output port C of the handler I/O is set to the input port, it reads out port information inputted to port C (C0 to C3).

Port information is input/output as 4-bit binary data, using C0 as LSB and C3 as MSB.

Variable

Parameter	*Value*
Description	Port information (output/input)
Data type	Long integer type (Long)
Range	0 to 15
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
SCPI.CONTrol.HANDler.C.MODE = "outp"
SCPI.CONTrol.HANDler.C.DATA = 8
```

```
Dim HdlCinp As Long
SCPI.CONTrol.HANDler.C.MODE = "inp"
HdlCinp = SCPI.CONTrol.HANDler.C.DATA
```

Related objects

SCPI.CONTrol.HANDler.C.MODE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler:C[:DATA] <numeric>

:CONTrol:HANDler:C[:DATA]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CONT:HAND:C:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:C 15"
```

```
10 OUTPUT 717;":CONT:HAND:C:MODE INP"
20 OUTPUT 717;":CONT:HAND:C?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.C.MODE

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.C.MODE = *Param*

Param = SCPI.CONTrol.HANDler.C.MODE

Description

This command sets/gets the input/output direction of port C of the handler I/O.

Variable

Parameter	*Param*
Description	Input/output direction of port C
Data type	Character string type (String)
Range	Select from the following: "INPut":Sets the port C to input. "OUTPut":Sets the port C to output.
Preset value	"INPut"

Examples

```
Dim HdlCmode As String
SCPI.CONTrol.HANDler.C.MODE = "OUTP"
HdlCmode = SCPI.CONTrol.HANDler.C.MODE
```

Related objects

SCPI.CONTrol.HANDler.C.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler:C:MODE {INPut|OUTPut}

:CONTrol:HANDler:C:MODE?

Query response

{INP|OUTP}<newline><^END>

Example of use

```
10 OUTPUT 717;":CONT:HAND:C:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:C:MODE?"
30 ENTER 717;A$
```

SCPI.CONTrol.HANDler.D.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.D.DATA = *Value* (for output port)

Value = SCPI.CONTrol.HANDler.D.DATA (for input port)

Description

When input/output port D of the handler I/O is set to the output port, it outputs port information to output port D (D0 to D3).

When input/output port D of the handler I/O is set to the input port, it reads out port information to input to port D (D0 to D3).

Port information is output as 4-bit binary data using D0 as LSB and D3 as MSB.

Variable

Parameter	*Value*
Description	Port information (output/input)
Data type	Long integer type (Long)
Range	0 to 15
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
SCPI.CONTrol.HANDler.D.MODE = "outp"
SCPI.CONTrol.HANDler.D.DATA = 8
```

```
Dim HdlDinp As Long
SCPI.CONTrol.HANDler.D.MODE = "inp"
HdlDinp = SCPI.CONTrol.HANDler.D.DATA
```

Related objects

SCPI.CONTrol.HANDler.D.MODE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler:D[:DATA] <numeric>

:CONTrol:HANDler:D[:DATA]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CONT:HAND:D:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:D 15"
```

```
10 OUTPUT 717;":CONT:HAND:D:MODE INP"
20 OUTPUT 717;":CONT:HAND:D?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.D.MODE

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.D.MODE = *Param*

Param = SCPI.CONTrol.HANDler.D.MODE

Description

This command sets/gets the input/output direction of port D of the handler I/O.

Variable

Parameter	*Param*
Description	Input/output direction of port D
Data type	Character string type (String)
Range	Select from the following: "INPut": Sets the port D to input. "OUTPut":Sets the port D to output.
Preset value	"INPut"

Examples

```
Dim HdlDmode As String
SCPI.CONTrol.HANDler.D.MODE = "OUTP"
HdlDmode = SCPI.CONTrol.HANDler.D.MODE
```

Related objects

SCPI.CONTrol.HANDler.D.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler:D:MODE {INPut|OUTPut}

:CONTrol:HANDler:D:MODE?

Query response

{INP|OUTP}<newline><^END>

Example of use

```
10 OUTPUT 717;":CONT:HAND:D:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:D:MODE?"
30 ENTER 717;A$
```

SCPI.CONTrol.HANDler.E.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.E.DATA = *Value* (for output)

Value = SCPI.CONTrol.HANDler.E.DATA (for input port)

Description

When input/output port E (port C + port D) of the handler I/O is set to the output port, it outputs port information to output port E (C0 to D3).

When input/output port E of the handler I/O is set to the input port, it reads out port information inputted to port E (C0 to D3).

Port information is output as 8-bit binary data using C0 as LSB and D3 as MSB.

Variable

Parameter	*Value*
Description	Port information (output/input)
Data type	Long integer type (Long)
Range	0 to 255
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
SCPI.CONTrol.HANDler.C.MODE = "outp"
SCPI.CONTrol.HANDler.D.MODE = "outp"
SCPI.CONTrol.HANDler.E.DATA = 128
```

```
Dim HdlEinp As Long
SCPI.CONTrol.HANDler.C.MODE = "inp"
SCPI.CONTrol.HANDler.D.MODE = "inp"
HdlEinp = SCPI.CONTrol.HANDler.E.DATA
```

Related objects

SCPI.CONTrol.HANDler.C.MODE

SCPI.CONTrol.HANDler.D.MODE

SCPI.CONTrol.HANDler.C.DATA

SCPI.CONTrol.HANDler.D.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler:E[:DATA] <numeric>

:CONTrol:HANDler:E[:DATA]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":CONT:HAND:C:MODE OUTP"
20 OUTPUT 717;":CONT:HAND:D:MODE OUTP"
30 OUTPUT 717;":CONT:HAND:E 128"
```

```
10 OUTPUT 717;":CONT:HAND:C:MODE INP"
20 OUTPUT 717;":CONT:HAND:D:MODE INP"
30 OUTPUT 717;":CONT:HAND:E?"
40 ENTER 717;A
```

SCPI.CONTrol.HANDler.EXTension.INDex.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.EXTension.INDex.STATe = *Status*

Status = SCPI.CONTrol.HANDler.EXTension.INDex.STATe

Description

Turns ON/OFF output of the INDEX signal to B6 of the handler I/O.

NOTE When you use port B6 as the output port, turn OFF the INDEX signal output. When output of the INDEX signal is turned ON, the bit 6 of the data output by the SCPI.CONTrol.HANDler.B.DATA object (the bit 14 of the data outputted by the SCPI.CONTrol.HANDler.F.DATA object) is ignored.

Variable

Parameter	*Status*
Description	ON/OFF of the INDEX signal output
Data type	Boolean type (Boolean)
Range	Select from the following: True or ON:Turns ON the INDEX signal output. False or OFF: Turns OFF the INDEX signal output.
Preset value	False or OFF

Examples

```
Dim Indx As Boolean
SCPI.CONTrol.HANDler.EXTension.INDex.STATe = 1
Indx = SCPI.CONTrol.HANDler.EXTension.INDex.STATe
```

Related objects

SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler[:EXTension]:INDex:STATe {ON|OFF|1|0}

:CONTrol:HANDler[:EXTension]:INDex:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CONT:HAND:IND:STAT ON"
20 OUTPUT 717;":CONT:HAND:IND:STAT?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe = *Status*

Status = SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe

Description

This command turns ON/OFF the output of READY FOR TRIGGER signal to B7 of the handler I/O.

NOTE When you use port B7 as the output port, turn OFF the READY FOR TRIGGER signal output. When outputting the READY FOR TRIGGER signal is turned ON, the bit 7of the data output by the SCPI.CONTrol.HANDler.B.DATA object (the bit 15 of the data output by the SCPI.CONTrol.HANDler.F.DATA object) is ignored.

Variable

Parameter	*Status*
Description	ON/OFF of the READY FOR TRIGGER signal output
Data type	Boolean type (Boolean)
Range	Select from the following: True or ON:Turns ON the READY FOR TRIGGER signal output. False or OFF:Turns OFF the READY FOR TRIGGER signal output.
Preset value	False or OFF

Examples

```
Dim RdyTrig As Boolean
SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe = 0
RdyTrig = SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe
```

Related objects

SCPI.CONTrol.HANDler.EXTension.INDex.STATe

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:CONTrol:HANDler[:EXTension]:RTRigger:STATe {ON|OFF|1|0}

:CONTrol:HANDler[:EXTension]:RTRigger:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":CONT:HAND:RTR:STAT ON"
20 OUTPUT 717;":CONT:HAND:RTR:STAT?"
30 ENTER 717;A
```

SCPI.CONTrol.HANDler.F.DATA

Object type

Method (**Write Only**)

Syntax

SCPI.CONTrol.HANDler.F.DATA = *Value*

Description

Outputs port information to output port F (port A + port B) of the handler I/O. Port information is output as 16-bit binary using A0 as LSB and B7 as MSB.

NOTE The bit 14 of the data output by this project is ignored when outputting the INDEX signal is turned ON (specifying True with the SCPI.CONTrol.HANDler.EXTension.INDex.STATe object).

NOTE The bit 15 of the data output by this project is ignored when outputting the READY FOR TRIGGER signal is turned ON (specifying True with the SCPI.CONTrol.HANDler.EXTension.RTRigger.STATe object).

Variable

Parameter	*Value*
Description	Port information (output)
Data type	Long integer type (Long)
Range	0 to 65535
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
SCPI.CONTrol.HANDler.F.DATA = 511
```

Related objects

SCPI.CONTrol.HANDler.A.DATA

SCPI.CONTrol.HANDler.B.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

```
:CONTrol:HANDler:F[:DATA] <numeric>
```

Example of use

```
10 OUTPUT 717;":CONT:HAND:F 511"
```

SCPI.CONTrol.HANDler.OUTPut(*Num*).DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.CONTrol.HANDler.OUTPut(*Num*).DATA = *Value*

Value = SCPI.CONTrol.HANDler.OUTPut(*Num*).DATA

Description

This command sets/gets data to OUTPUT1 or OUTPUT2 of the handler I/O.

Variable

Parameter	*Num*
Description	Number of the OUTPUT terminal
Data type	Long integer type (Long)
Range	1 or 2
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Parameter	*Value*
Description	Polarity (High/Low)
Data type	Long integer type (Long)
Range	Select from the following: • 1: Specifies LOW. • 0: Specifies HIGH.

Examples

```
Dim HdlPol As Long
SCPI.CONTrol.HANDler.OUTPut(1).DATA = 1
HdlPol = SCPI.CONTrol.HANDler.OUTPut(1).DATA
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

```
:CONTrol:HANDler:OUTPut{[1]|2}[:DATA] {1|0}
:CONTrol:HANDler:OUTPut{[1]|2}[:DATA]?
```

Query response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":CONT:HAND:OUTP1 1"
20 OUTPUT 717;":CONT:HAND:OUTP1?"
30 ENTER 717;A
```

DISPLAY

SCPI.DISPlay.ANNotation.FREQuency.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.ANNotation.FREQuency.STATe = *Status*

Status = SCPI.DISPlay.ANNotation.FREQuency.STATe

Description

This command turns ON/OFF the frequency display on the LCD display.

Variable

Parameter	*Status*
Description	Sets/Gets ON/OFF state of the frequency display
Data type	Boolean type (Boolean)
Range	Select either of the following: • True or ON: Turns ON the frequency display. • False or OFF: Turns OFF the frequency display.
Preset value	True or ON

Examples

```
Dim DispFreq As Boolean
SCPI.DISPlay.ANNotation.FREQuency.STATe = 0
DispFreq = SCPI.DISPlay.ANNotation.FREQuency.STATe
```

Equivalent key

Display > **Frequency**

Equivalent SCPI command

Syntax

:DISPlay:ANNotation:FREQuency[:STATe] {ON|OFF|1|0}

:DISPlay:ANNotation:FREQuency[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:ANN:FREQ OFF"
20 OUTPUT 717;":DISP:ANN:FREQ?"
30 ENTER 717;A
```

SCPI.DISPlay.CCLear

Object type

Method (**Write Only**)

Syntax

SCPI.DISPlay.CCLear

Description

This command clears the error message displayed in the status bar (at the bottom of the LCD display).

Examples

SCPI.DISPlay.CCLear

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:DISPlay:CCLear

Example of use

```
10 OUTPUT 717;":DISP:CCL"
```

SCPI.DISPlay.CLOCk

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.CLOCk = *Status*

Status = SCPI.DISPlay.CLOCk

Description

This command turns ON/OFF the clock display in the instrument status bar (at the right bottom of the LCD display).

Variable

Parameter	*Status*
Description	ON/OFF of the clock display
Data type	Boolean type (Boolean)
Range	Select either of the following: • True or ON: Turns ON the clock display. • False or OFF: Turns OFF the clock display.
Preset value	True or ON

Examples

```
Dim DispTime As Boolean
SCPI.DISPlay.CLOCk = ON
DispTime = SCPI.DISPlay.CLOCk
```

Equivalent key

System > **Misc Setup** > **Clock Setup** > **Show Clock**

Equivalent SCPI command

Syntax

:DISPlay:CLOCk {ON|OFF|1|0}

:DISPlay:CLOCk?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:CLOC OFF"
20 OUTPUT 717;":DISP:CLOC?"
30 ENTER 717;A
```

SCPI.DISPlay.COLor(*Dnum*).BACK

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.COLor(*Dnum*).BACK = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).BACK

Description

This command sets/gets the background color for normal display (*Dnum*:1) and inverted display (Dnum:2).

Variable

Parameter	*Dnum*
Description	Select either of the following: 1: Normal display 2: Inverted display
Data type	Long integer type (Long)
Range	1 or 2
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when the command is executed.

Parameter	*Data*
Description	Indicates 3-element array data. • *Data(0) :* Sets amount of red. • *Data(1) :* Sets amount of green. • *Data(2) :* Sets amount of blue. The index of the array starts from 0.
Data type	Variant type (Variant)

Range	• *Data(0)* 0 to 5 • *Data(1)* 0 to 5 • *Data(2)* 0 to 5
Resolution	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim BackColor As Variant
SCPI.DISPlay.COLor(1).BACK = Array(1,2,3)
BackColor = SCPI.DISPlay.COLor(1).BACK
```

Related objects

SCPI.DISPlay.COLor(Dnum).RESet

Equivalent key

System > **Misc Setup** > **Display Setup** > **Color Setup** > **Normal|Invert** > **Background**

Equivalent SCPI command

Syntax

:DISPlay:COLor{[1]|2}:BACK <numeric 1>,<numeric 2>,<numeric 3>

:DISPlay:COLor{[1]|2}:BACK?

Query response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:COL:BACK 1,2,3"
20 OUTPUT 717;":DISP:COL:BACK?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.COLor(*Dnum*).GRATicule(*Gnum*)

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.COLor(*Dnum*).GRATicule(*Gnum*) = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).GRATicule(*Gnum*)

Description

This command sets/gets:

1. Color of the graticule label.
2. Outer frame line of the graph (*Gnum*:1).
3. Color of the grid line of the graph (*Gnum*:2).

for the normal display (*Dnum*:1) and inverted display (*Dnum*:2).

Variable

Parameter	*Gnum*
Description	The number of items: 1: The outer frame line of the graph 2: The color of the grid line of the graph
Data type	Long integer type (Long)
Range	1 to 2
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Parameter	*Data*
Description	Indicates 3-element array data. • *Data(0) :* Sets amount of red. • *Data(1) :* Sets amount of green.

	• *Data(2) :* Sets amount of blue. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	• *Data(0) :* 0 to 5 • *Data(1) :* 0 to 5 • *Data(2) :* 0 to 5
Resolution	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim GritColor As Variant
SCPI.DISPlay.COLor(1).GRATicule(1) = Array(1,2,3)
GritColor = SCPI.DISPlay.COLor(1).GRATicule(1)
```

Related objects

SCPI.DISPlay.COLor(Dnum).RESet

Equivalent key

System > **Misc Setup** > **Display Setup** > **Color Setup** > **Normal**|**Invert** > **Graticule Main**|**Graticule Sub**

Equivalent SCPI command

Syntax

:DISPlay:COLor{[1]|2}:GRATicule{[1]|2} <numeric 1>,<numeric 2>,<numeric 3>

:DISPlay:COLor{[1]|2}:GRATicule{[1]|2}?

Query response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:COL1:GRAT1 1,2,3"
20 OUTPUT 717;":DISP:COL1:GRAT1?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.COLor(*Dnum*).LIMit(*Lnum*)

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.COLor(*Dnum*).LIMit(*Lnum*) = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).LIMit(*Lnum*)

Description

This command sets/gets:

- Fail display color used for the limit test result, Bandwidth test result and Ripple test result (*Lnum*:1)
- Color of the limit line (*Lnum*:2)
  for normal display (*Dnum*:1) and inverted display (*Dnum*:2).

Variable

Parameter	*Lnum*
Description	The number of item 1: The limit test result (Fail/Pass) 2: The limit line
Data type	Long integer type (Long)
Range	1 to 2
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Parameter	*Data*
Description	Indicates 3-element array data. • *Data(0) :* Sets amount of red. • *Data(1) :* Sets amount of green.

	• *Data(2) :* Sets amount of blue. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	• *Data(0) :* 0 to 5 • *Data(1) :*0 to 5 • *Data(2) :*0 to 5
Resolution	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim LimColor As Variant
SCPI.DISPlay.COLor(1).LIMit(1) = Array(1,2,3)
LimColor = SCPI.DISPlay.COLor(1).LIMit(1)
```

Related objects

SCPI.DISPlay.COLor(Dnum).RESet

Equivalent key

System > **Misc Setup** > **Color Setup** > **Normal**|**Invert** > **Limit Fail**|**Limit Line**

Equivalent SCPI command

Syntax

:DISPlay:COLor{[1]|2}:LIMit{[1]|2} <numeric 1>,<numeric 2>,<numeric 3>

:DISPlay:COLor{[1]|2}:LIMit{[1]|2}?

Query response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:COL1:LIM1 1,2,3"
20 OUTPUT 717;":DISP:COL1:LIM1?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.COLor(*Dnum*).RESet

Object type

Method (**Write-only**)

Syntax

SCPI.DISPlay.COLor(*Dnum*).RESet

Description

This command resets the display color settings for all the items to the factory preset state, for normal display (*Dnum*:1) and inverted display (*Dnum*:2).

Examples

SCPI.DISPlay.COLor(1).RESet

Related objects

SCPI.DISPlay.COLor(Dnum).BACK

SCPI.DISPlay.COLor(Dnum).GRATicule(Gnum)

SCPI.DISPlay.COLor(Dnum).LIMit(Lnum)

SCPI.DISPlay.COLor(Dnum).TRACe(Tr).DATA

SCPI.DISPlay.COLor(Dnum).TRACe(Tr).MEMory

Equivalent key

System > **Misc Setup** > **Color Setup** > **Normal**|**Invert** > **Reset Color** > **OK**

Equivalent SCPI command

Syntax

:DISPlay:COLor{[1]|2}:RESet

Example of use

```
10 OUTPUT 717;":DISP:COL1:RES"
```

SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).DATA = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).DATA

Description

This command sets/gets the color of selected trace (*Tr*), for normal display (*Dnum*:1) and inverted display (*Dnum*:2).

Variable

Parameter	*Data*
Description	Indicates 3-element array data. • *Data(0) :*Sets amount of red. • *Data(1) :*Sets amount of green. • *Data(2) :*Sets amount of blue. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	• *Data(0) :* 0 to 5 • *Data(1) :* 0 to 5 • *Data(2) :* 0 to 5
Resolution	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim TrColor As Variant
SCPI.DISPlay.COLor(1).TRACe(1).DATA = Array(1,2,3)
TrColor = SCPI.DISPlay.COLor(1).TRACe(1).DATA
```

Related objects

SCPI.DISPlay.COLor(Dnum).RESet

Equivalent key

System > **Misc Setup** > **Color Setup** > **Normal**|**Invert** > **Data Trace X** (X=1 to 4)

Equivalent SCPI command

Syntax

:DISPlay:COLor{[1]|2}:TRACe{[1]-4}:DATA <numeric 1>,<numeric 2>,<numeric 3>

:DISPlay:COLor{[1]|2}:TRACe{[1]-4}:DATA?

Query response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:COL1:TRAC1:DATA 1,2,3"
20 OUTPUT 717;":DISP:COL1:TRAC1:DATA?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).MEMory

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).MEMory = *Data*

Data = SCPI.DISPlay.COLor(*Dnum*).TRACe(*Tr*).MEMory

Description

This command sets/gets the color of the memory trace for the selected (*Tr*), for normal display (*Dnum*:1) and inverted display (*Dnum*:2).

Variable

Parameter	*Data*
Description	Indicates 3-element array data. • *Data(0) :*Sets amount of red. • *Data(1) :*Sets amount of green. • *Data(2) :*Sets amount of blue. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	• *Data(0) :*0 to 5 • *Data(1) :*0 to 5 • *Data(2) :*0 to 5
Resolution	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim TrColor As Variant
SCPI.DISPlay.COLor(1).TRACe(1).MEMory = Array(1,2,3)
TrColor = SCPI.DISPlay.COLor(1).TRACe(1).MEMory
```

Related objects

SCPI.DISPlay.COLor(Dnum).RESet

Equivalent key

System > **Misc Setup** > **Color Setup** > **Normal**|**Invert** > **Mem Trace X** (X=1 to 4)

Equivalent SCPI command

Syntax

:DISPlay:COLor{[1]|2}:TRACe{[1]-4}:MEMory <numeric 1>,<numeric 2>,<numeric 3>

:DISPlay:COLor{[1]|2}:TRACe{[1]-4}:MEMory?

Query response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:COL1:TRAC1:MEM 1,2,3"
20 OUTPUT 717;":DISP:COL1:TRAC1:MEM?"
30 ENTER 717;A,B,C
```

SCPI.DISPlay.ECHO.CLEar

Object type

Method (**Write Only**)

Syntax

SCPI.DISPlay.ECHO.CLEar

Description

This command clears all character strings displayed in the echo window.

Examples

SCPI.DISPlay.ECHO.CLEar

Related objects

SCPI.DISPlay.ECHO.DATA

Equivalent key

Macro Setup > **Clear Echo**

Equivalent SCPI command

Syntax

:DISPlay:ECHO:CLEar

Example of use

```
10 OUTPUT 717;":DISP:ECHO:CLE"
```

SCPI.DISPlay.ECHO.DATA

Object type

Property (**Write Only**)

Syntax

SCPI.DISPlay.ECHO.DATA = *Cont*

Description

This command displays a character string in the echo window. This command is different from ECHO command as it displays a single character string.

Variable

Parameter	*Cont*
Description	String you want to display in the echo window.
Data type	Character string type (String)
Range	254 characters or less

Examples

```
SCPI.DISPlay.ECHO.DATA = "Test Result"
SCPI.DISPlay.TABLe.TYPE = "echo"
SCPI.DISPlay.TABLe.STATe = True
```

Related objects

ECHO

SCPI.DISPlay.TABLe.TYPE

SCPI.DISPlay.TABLe.STATe

SCPI.DISPlay.ECHO.CLEar

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:DISPlay:ECHO[:DATA] <string>

Example of use

```
10 OUTPUT 717;":DISP:ECHO ""TEST RESULT"""
```

SCPI.DISPlay.ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.ENABle = *Status*

Status = SCPI.DISPlay.ENABle

Description

This command turns ON/OFF the display update on the E5061B measurement screen.

Variable

Parameter	*Status*
Description	ON/OFF of the display update of the E5061B measurement screen
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the display update. • False or OFF : Turns OFF the display update.
Preset value	True or ON

Examples

```
Dim DispUpdt As Boolean
SCPI.DISPlay.ENABle = False
DispUpdt = SCPI.DISPlay.ENABle
```

Equivalent key

Display > **Update**

Equivalent SCPI command

Syntax

:DISPlay:ENABle {ON|OFF|1|0}

:DISPlay:ENABle?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:ENAB OFF"
20 OUTPUT 717;":DISP:ENAB?"
30 ENTER 717;A
```

SCPI.DISPlay.FSIGn

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.FSIGn = *Status*

Status = SCPI.DISPlay.FSIGn

Description

This command turns ON/OFF the "Fail" display on the LCD screen when the limit test, bandwidth test and ripple test fails.

Variable

Parameter	*Status*
Description	ON/OFF of the "Fail" display when the limit test fails
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the "Fail" display. • False or OFF : Turns OFF the "Fail" display.
Preset value	True or ON

ON/OFF of the Fail display cannot be set at each test. When the Fail display of either of test is turned ON, the Fail display of other tests turns ON, too.

Examples

```
Dim DispFail As Boolean
SCPI.DISPlay.FSIGn = False
DispFail = SCPI.DISPlay.FSIGn
```

Related objects

SCPI.CALCulate(Ch).SELected.LIMit.STATe

SCPI.CALCulate(Ch).SELected.RLIMit.STATe

SCPI.CALCulate(Ch).SELected.BLIMit.STATe

Equivalent key

Analysis > **Limit Test > Fail Sign**

Analysis > **Ripple Limit > Fail Sign**

Analysis > **Bandwidth limit > Fail Sign**

Equivalent SCPI command

Syntax

:DISPlay:FSIGn {ON|OFF|1|0}

:DISPlay:FSIGn?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:FSIG OFF"
20 OUTPUT 717;":DISP:FSIG?"
30 ENTER 717;A
```

SCPI.DISPlay.IMAGe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.IMAGe = *Param*

Param = SCPI.DISPlay.IMAGe

Description

This command sets/gets the display type of the LCD display.

Variable

Parameter	*Param*
Description	Display type of the LCD display
Data type	Character string type (String)
Range	Select from the following: • "NORMal" : Specifies the normal display (background color: black). • "INVert": Specifies the display in which the color of the normal display is inversed (background color: white).
Preset value	"NORMal"

Examples

```
Dim DispImg As String
SCPI.DISPlay.IMAGe = "inv"
DispImg = SCPI.DISPlay.IMAGe
```

Equivalent key

Display > **Invert Color**

Equivalent SCPI command

Syntax

:DISPlay:IMAGe {NORMal|INVert}

:DISPlay:IMAGe?

Query response

{NORM|INV}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:IMAG INV"
20 OUTPUT 717;":DISP:IMAG?"
30 ENTER 717;A$
```

SCPI.DISPlay.MAXimize

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.MAXimize = *Status*

Status = SCPI.DISPlay.MAXimize

Description

This command turns ON/OFF the window maximization of the active channel.

If the maximization is set to ON, only the window of the active channel is maximized on the LCD display and the windows of the other channels are not displayed.

Variable

Parameter	*Status*
Description	ON/OFF of the window maximization
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the window maximization. • False or OFF: Turns OFF the window maximization.
Preset value	False or OFF

Examples

```
Dim ChMax As Boolean
SCPI.DISPlay.SPLit = "d1_2"
SCPI.DISPlay.WINDow(2).ACTivate
SCPI.DISPlay.MAXimize = True
ChMax = SCPI.DISPlay.MAXimize
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

Equivalent key

Channel Max

Equivalent SCPI command

Syntax

:DISPlay:MAXimize {ON|OFF|1|0}

:DISPlay:MAXimize?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:MAX ON"
20 OUTPUT 717;":DISP:MAX?"
30 ENTER 717;A
```

SCPI.DISPlay.SKEY.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.SKEY.STATe = *Status*

Status = SCPI.DISPlay.SKEY.STATe

Description

This command turns ON/OFF the display of the softkey menu bar.

Variable

Parameter	*Status*
Description	ON/OFF of the softkey menu bar display
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the softkey menu bar display. • False or OFF: Turns OFF the softkey menu bar display.
Preset value	True or ON

Examples

```
Dim DispSkey As Boolean
SCPI.DISPlay.SKEY.STATe = False
DispSkey = SCPI.DISPlay.SKEY.STATe
```

Equivalent key

Entry Off

Equivalent SCPI command

Syntax

:DISPlay:SKEY[:STATe] {ON|OFF|1|0}

:DISPlay:SKEY[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:SKEY ON"
20 OUTPUT 717;":DISP:SKEY?"
30 ENTER 717;A
```

SCPI.DISPlay.SPLit

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.SPLit = *Param*

Param = SCPI.DISPlay.SPLit

Description

Sets the layout of the channel windows on the LCD display.

Variable

Parameter	*Param*
Description	Layout of channel windows
Data type	Character string type (String)
Range	Select from the following: • "D1" • "D12" • "D1_2" • "D112" • "D1_1_2" • "D123" • "D1_2_3" • "D12_33" • "D11_23" • "D13_23" • "D12_13" • "D1234" • "D1_2_3_4" • "D12_34" Refer to link " Window Graph layouts and command parameters" for the confirmation of the window layout.

Preset value	"D1"

Examples

```
Dim ChanAloc As String
SCPI.DISPlay.SPLit = "d12_34"
ChanAloc = SCPI.DISPlay.SPLit
```

Related objects

SCPI.DISPlay.WINDow(Ch).SPLit

Equivalent key

Display > **Allocate Channels**

Equivalent SCPI command

Syntax

```
:DISPlay:SPLit
{D1|D12|D1_2|D112|D1_1_2|D123|D1_2_3|D12_33|D11_23|D13_23|D12_13| D1234|D1_2_3_4|D12_34}
:DISPlay:SPLit?
```

Query response

```
{D1|D12|D1_2|D112|D1_1_2|D123|D1_2_3|D12_33|D11_23|D13_23|D12_13| D1234|D1_2_3_4|D12_34}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":DISP:SPL D1_2"
20 OUTPUT 717;":DISP:SPL?"
30 ENTER 717;A$
```

SCPI.DISPlay.TABLe.POSition.RECTangle

Object type

Property (**Read Only**)

Syntax

Param = SCPI.DISPlay.TABLe.POSition.RECTangle

Description

This command reads the display coordinates position of Table area (the top left of the display is [0, 0]). If SCPI.DISPlay.TABLe.STATe is OFF, 0, 0, 0, 0 are returned. .

Variable

Parameter	*Param*
Description	Indicates the coordinates position of Table Area. • *Param(0)* : coordinates X position of top left of Table Area. • *Param(1)* : coordinates Y position of top left of Table Area. • *Param(2)* : coordinates X position of bottom right of Table Area. • *Param(3)* : coordinates Y position of bottom right of Table Area. The index of the array starts from 0.
Data type	Variant type (Variant)
Resolution	1

Examples

```
Dim TablePos() As Variant
SCPI.DISPlay.TABLe.STATe = True
TablePos = SCPI.DISPlay.TABLe.POSition.RECTangle
```

Related objects

SCPI.DISPlay.TABLe.STATe

Equivalent key

None

Equivalent SCPI command

Syntax

:DISPlay:TABLe:POSition[:RECTangle]?

Query response

{numeric1},{numeric2},{numeric3},{numeric4},<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:TABL:STAT ON"
20 OUTPUT 717;":DISP:TABL:POS?"
20 ENTER 717;A, B, C, D
```

SCPI.DISPlay.TABLe.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.TABLe.STATe = *Status*

Status = SCPI.DISPlay.TABLe.STATe

Description

This command turns ON/OFF the display of the window that appears in the lower part of the LCD display (specified by SCPI.DISPlay.TABLe.TYPE object).

Variable

Parameter	*Status*
Description	ON/OFF of the display of the window that appears in the lower part of the LCD display
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the display. • False or OFF: Turns OFF the display.
Preset value	False or OFF

Examples

```
Dim DispTbl As Boolean
SCPI.DISPlay.TABLe.TYPE = "echo"
SCPI.DISPlay.TABLe.STATe = True
DispTbl = SCPI.DISPlay.TABLe.STATe
```

Related objects

SCPI.DISPlay.TABLe.TYPE

Equivalent key

Sweep Setup > **Edit Segment Table**

Marker Fctn > **Marker Table**

Analysis > **Limit Test** > **Edit Limit Line**

Analysis > **Ripple Limit** > **Edit Ripple Line**

Macro Setup > **Echo Window**

Cal > **Power Calibration** > **Loss Compen**

Cal > **Power Calibration** > **Sensor A Settings | Sensor B Settings**

When performing the operation from the front panel, you select the type of the window that appears in the lower part of the LCD display and turn ON/OFF the display at the same time.

Equivalent SCPI command

Syntax

:DISPlay:TABLe[:STATe] {ON|OFF|1|0}

:DISPlay:TABLe[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:TABL ON"
20 OUTPUT 717;":DISP:TABL?"
30 ENTER 717;A
```

SCPI.DISPlay.TABLe.TYPE

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.TABLe.TYPE = *Param*

Param = SCPI.DISPlay.TABLe.TYPE

Description

This command selects the type of the window that appears in the lower part of the LCD display.

Variable

Parameter	*Param*
Description	Window type
Data type	Character string type (String)
Range	Select from the following: • "MARKer": Specifies the marker table window. • "LIMit": Specifies the limit test table window. • "SEGMent": Specifies the segment table window. • "ECHO": Specifies the echo window. • "RLIMit": Specifies the ripple test table window.
Preset value	"MARKer"

Examples

```
Dim TblType As String
SCPI.DISPlay.TABLe.TYPE = "echo"
SCPI.DISPlay.TABLe.STATe = True
TblType = SCPI.DISPlay.TABLe.TYPE
```

Related objects

SCPI.DISPlay.TABLe.STATe

Equivalent key

Sweep Setup > **Edit Segment Table**

Marker Fctn > **Marker Table**

Analysis > **Limit Test** > **Edit Limit Line**

Analysis > **Ripple Limit** > **Edit Ripple Line**

Macro Setup > **Echo Window**

When performing the operation from the front panel, you select the type of the window that appears in the lower part of the LCD display and turn ON/OFF the display at the same time.

Equivalent SCPI command

Syntax

:DISPlay:TABLe:TYPE {MARKer|LIMit|SEGMent|ECHO|RLIMit}

:DISPlay:TABLe:TYPE?

Query response

{MARK|LIM|SEGM|ECHO|RLIM}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:TABL:TYPE SEGM"
20 OUTPUT 717;":DISP:TABL:TYPE?"
30 ENTER 717;A$
```

SCPI.DISPlay.UPDate.IMMediate

Object type

Method (**Write-only**)

Syntax

SCPI.DISPlay.UPDate.IMMediate

Description

This command executes the display update once when the display update of the LCD screen is set to OFF (specifying False with the SCPI.DISPlay.ENABle object).

Examples

```
SCPI.DISPlay.ENABle = False
SCPI.DISPlay.UPDate.IMMediate
```

Related objects

SCPI.DISPlay.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:DISPlay:UPDate[:IMMediate]

Example of use

```
10 OUTPUT 717;":DISP:UPD"
```

SCPI.DISPlay.WINDow(*Ch*).ACTivate

Object type

Method (**Write-only**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).ACTivate

Description

This command specifies selected channel (*Ch*) as the active channel.

NOTE You can set only a channel displayed to the active channel. If this object is used to set a channel not displayed to the active channel, an error occurs when executed and the object is ignored.

Examples

```
SCPI.DISPlay.SPLit = "d1_2"
SCPI.DISPlay.WINDow(2).ACTivate
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Channel Prev / **Channel Next**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:ACTivate

Example of use

```
10 OUTPUT 717;":DISP:WIND1:ACT"
```

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.ALIGn.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.ALIGn.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.ALIGn.STATe

Description

This command turn ON/OFF the mode that align the marker display position of each trace based on trace 1, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the mode that align the marker display position of each trace based on trace 1
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON:Turns ON the mode that allign marker display position based on trace 1. • False or OFF:Turns OFF the alignment.
Preset value	True or ON

Examples

```
Dim AnnMarkAlig As Boolean
SCPI.DISPlay.WINDow(1).ANNotation.MARKer.ALIGn.STATe = False
AnnMarkAlig = SCPI.DISPlay.WINDow(1).ANNotation.MARKer.ALIGn.STATe
```

Related objects

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle.STATe

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.X

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.Y

Equivalent key

Marker Fctn > **Annotation Options** > **Align**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:ANNotation:MARKer:ALIGn[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-4}:ANNotation:MARKer:ALIGn[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:ANN:MARK:ALIG OFF"
20 OUTPUT 717;":DISP:WIND1:ANN:MARK:ALIG?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.SINGle.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.SINGle.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKer.SINGle.STATe

Description

This command turns ON/OFF the display of the marker value of only active traces, for the selected channel (*Ch*).

NOTE If the function is turned OFF, marker values of all traces (markers) are displayed.

Variable

Parameter	*Status*
Description	ON/OFF the display of the marker value of only active
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Displays the marker values of only the active traces.(ON) • False or OFF: Displays the marker values of all the traces. (OFF)
Preset value	True or ON

Examples

```
Dim AnnMarkAlig As Boolean
SCPI.DISPlay.WINDow(1).ANNotation.MARKer.SINGle.STATe = False
AnnMarkAlig = SCPI.DISPlay.WINDow(1).ANNotation.MARKer.SINGle.STATe
```

Related objects

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn.STATe

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.X

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.Y

Equivalent key

Marker Fctn > **Annotation Options** > **Active Only**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:ANNotation:MARKer:SINGle[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-4}:ANNotation:MARKer:SINGle[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:ANN:MARK:SING OFF"
20 OUTPUT 717;":DISP:WIND1:ANN:MARK:SING?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).LABel

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).LABel = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).LABel

Description

This command turns ON/OFF the graticule label display of the graph of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the graticule label display of the graph
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the graticule label display. • False or OFF: Turns OFF the graticule label display.
Preset value	True or ON

Examples

```
Dim DispGrat As Boolean
SCPI.DISPlay.WINDow(1).LABel = False
DispGrat = SCPI.DISPlay.WINDow(1).LABel
```

Equivalent key

Display > **Graticule Label**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:LABel {ON|OFF|1|0}

:DISPlay:WINDow{[1]-4}:LABel?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:LAB ON"
20 OUTPUT 717;":DISP:WIND1:LAB?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).MAXimize

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).MAXimize = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).MAXimize

Description

This command turns ON/OFF the maximization of the active trace of selected channel (*Ch*).

NOTE If you turned ON the maximization, only the maximized active trace is displayed in the window and the other traces are not displayed.

Variable

Parameter	*Status*
Description	ON/OFF of the maximization of the active trace
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the maxim display. • False or OFF: Turns OFF the maxim display.
Preset value	False or OFF

Examples

```
Dim TracMax As Boolean
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.DISPlay.WINDow(1).MAXimize = True
TracMax = SCPI.DISPlay.WINDow(1).MAXimize
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.DISPlay.MAXimize

Equivalent key

Trace Max

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:MAXimize {ON|OFF|1|0}

:DISPlay:WINDow{[1]-4}:MAXimize?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:MAX ON"
20 OUTPUT 717;":DISP:WIND1:MAX?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).SPLit

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).SPLit = *Param*

Param = SCPI.DISPlay.WINDow(*Ch*).SPLit

Description

This command sets/gets the graph layout of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Graph layout
Data type	Character string type (String)
Range	Select from the following: • "D1" • "D12" • "D1_2" • "D112" • "D1_1_2" • "D123" • "D1_2_3" • "D12_33" • "D11_23" • "D13_23" • "D12_13" • "D1234" • "D1_2_3_4" • "D12_34" Refer to link " Window Graph layouts and command parameters" for the confirmation of the channel window layout.

Preset value	"D1"

Examples

```
Dim TracAloc As String
SCPI.DISPlay.WINDow(1).SPLit = "d1_2"
TracAloc = SCPI.DISPlay.WINDow(1).SPLit
```

Related objects

SCPI.DISPlay.SPLit

Equivalent key

Display > **Allocate Traces**

Equivalent SCPI command

Syntax

```
:DISPlay:WINDow{[1]-4}:SPLit {D1|D12|D1_2|D112|D1_1_2|
D123|D1_2_3|D12_33|D11_23|D13_23|D12_13|D1234|D1_2_3_4|D12_3
4|
D1X1}
:DISPlay:WINDow{[1]-4}:SPLit?
```

Query response

```
{D1|D12|D1_2|D112|D1_1_2|D123|D1_2_3|D12_33|D11_23|D13_23|
D12_13|D1234|D1_2_3_4|D12_34}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":DISP:WIND:SPL D1_2"
20 OUTPUT 717;":DISP:WIND:SPL?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).TITLe.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TITLe.DATA = *Lbl*

Lbl = SCPI.DISPlay.WINDow(*Ch*).TITLe.DATA

Description

This command sets/gets the title label displayed in the title area of the selected channel (*Ch*).

Variable

Parameter	*Lbl*
Description	Title label
Data type	Character string type (String)
Range	254 characters or less
Preset value	""

Examples

```
Dim TtlLbl As String
SCPI.DISPlay.WINDow(1).TITLe.DATA = "Filter"
SCPI.DISPlay.WINDow(1).TITLe.STATe = True
TtlLbl = SCPI.DISPlay.WINDow(1).TITLe.DATA
```

Related objects

SCPI.DISPlay.WINDow(Ch).TITLe.STATe

Equivalent key

Display > **Edit Title Label**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TITLe:DATA <string>

:DISPlay:WINDow{[1]-4}:TITLe:DATA?

Query response

{string}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TITL:DATA ""Title"""
20 OUTPUT 717;":DISP:WIND1:TITL?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).TITLe.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TITLe.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).TITLe.STATe

Description

This command turns ON/OFF the title label display in the title area of channels 1 to 16 (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the title label display
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the title label display. • False or OFF: Turns ON the title label display.
Preset value	False or OFF

Examples

```
Dim DispTtl As Boolean
SCPI.DISPlay.WINDow(1).TITLe.DATA = "Filter"
SCPI.DISPlay.WINDow(1).TITLe.STATe = True
DispTtl = SCPI.DISPlay.WINDow(1).TITLe.STATe
```

Related objects

SCPI.DISPlay.WINDow(Ch).TITLe.DATA

Equivalent key

Display > **Title Label**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TITLe[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-4}:TITLe[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TITL ON"
20 OUTPUT 717;":DISP:WIND1:TITL?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.X

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.X = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.X

Description

This command sets/gets the display position of the marker value on the X-axis by a percentage of a width of the display span, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Display position of the marker value on the X-axis.
Data type	Long integer type (Long)
Range	-15 to 100
Preset value	1
Unit	% (percent)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim AnnMPosX As Long
SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.X = 15
AnnMPosX = SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.X
```

Related objects

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn.STATe

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle.STATe

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.Y

Equivalent key

Marker Fctn > **Annotation Options** > **Marker Info X Pos**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:ANNotation:MARKer:POSition:X <numeric>

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:ANNotation:MARKer:POSition:X?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:X 33"
20 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:X?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.Y

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.Y = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition.X

Description

This command sets/gets the display position of the marker value on the X-axis by a percentage of a width of the display span, for the selected trace (*Tr*) of the selected channel (*Ch*), and the marker value on Y axis by a percentage of a height of the display span.

Variable

Parameter	*Value*
Description	Display position of the marker value on the Y-axis.
Data type	Long integer type (Long)
Range	-15 to 100
Preset value	1
Unit	% (percent)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim AnnMPosY As Long
SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.Y = 23
AnnMPosY = SCPI.DISPlay.WINDow(1).TRACe(1).ANNotation.MARKer.POSition.Y
```

Related objects

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn.STATe

SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle.STATe

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).ANNotation.MARKer.POSition.X

Equivalent key

Marker Fctn > **Annotation Options** > **Marker Info Y Pos**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:ANNotation:MARKer:POSition:Y <numeric>

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:ANNotation:MARKer:POSition:Y?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:Y 33"
20 OUTPUT 717;":DISP:WIND1:TRAC1:ANN:MARK:POS:Y?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.YAXis.MODE

Object Type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.YAXis.MODE = *Param*

Param = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.YAXis.MODE

Description

This command sets/gets the color mode for the Y-axis labels, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The color mode for the Y-axis labels
Data Type	Character string type (String)
Range	Select from the following: • "AUTO" : Specifies the same color as the Graticule Main color (default color is R:3 G:3 B:3. • "RELative": Specifies the same color as the Data Trace color (The default color is different at each trace).
Preset Value	"AUTO"

Examples

```
Dim Ylbl_Color as String
SCPI.DISPlay.WINDow(1).TRACe(4).ANNotation.YAXis.MODE = "REL"
Ylbl_Color = SCPI.DISPlay.WINDow(1).TRACe(4).ANNotation.YAXis.MODE
```

Related Objects

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:ANNotation:YAXis:MODE {AUTO|RELative}

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:ANNotation:YAXis:MODE?

Query Response

{AUTO|REL} <newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND2:TRAC1:ANN:YAX:MODE AUTO"
20 OUTPUT 717;":DISP:WIND2:TRAC1:ANN:YAX:MODE?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).MEMory. STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).MEMory.STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).MEMory.STATe

Description

This command turns ON/OFF the memory trace display, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the memory trace display
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the memory trace display. • False or OFF: Turns OFF the memory trace display.
Preset value	False or OFF

Examples

```
Dim DispMem As Boolean
SCPI.DISPlay.WINDow(1).TRACe(2).MEMory.STATe = True
DispMem = SCPI.DISPlay.WINDow(1).TRACe(2).MEMory.STATe
```

Related objects

SCPI.CALCulate(Ch).SELected.MATH.MEMorize

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).STATe

Equivalent key

Display > **Display** > **Mem** (when the data trace display is OFF)

Display > **Display** > **Data & Mem** (when the data trace display is ON)

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:MEMory[:STATe] {ON|OFF|1|0}

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:MEMory[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:MEM ON"
20 OUTPUT 717;":DISP:WIND1:TRAC1:MEM?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).STATe = *Status*

Status = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).STATe

Description

This command turns ON/OFF the data trace display, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the data trace display
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the data trace display. • False or OFF: Turns OFF the data trace display.
Preset value	True or ON

Examples

```
Dim DispTrac As Boolean
SCPI.DISPlay.WINDow(1).TRACe(2).STATe = False
DispTrac = SCPI.DISPlay.WINDow(1).TRACe(2).STATe
```

Related objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).MEMory.STATe

Equivalent key

Display > **Display** > **Data** (when the memory trace display is OFF)

Display > **Display** > **Data & Mem** (when the memory trace display is ON)

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :STATe {ON|OFF|1|0}

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:STAT ON"
20 OUTPUT 717;":DISP:WIND1:TRAC1:STAT?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.ABSolute.UNIT

Object Type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.ABSolute.UNIT = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.ABSolute.UNIT

Description

This command defines or gets Y-Axis unit in absolute format (LIN-MAG, REAL or IMAG). This is applicable for impedance measurement as well as network measurement. By default, the unit is "Units".Use this command to define other units to display "Ω", "F", "H" etc. in impedance measurement.

Variable

Parameter	*Value*
Description	Y-Axis
Data Type	Character string type (String)
Range	254 chars
Preset Value	"Units"
Unit	-
Resolution	-

Examples

```
Dim OriginalUnit as String
Dim NewUnit as String
Dim TestUnit as String
OriginalUnit = SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.ABSolute.UNIT
'*Return value = Units

NewUnit ="abc"
SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.ABSolute.UNIT = NewUnit
TestUnit = SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.ABSolute.UNIT
'*Return value = abc
```

Related Objects

SCPI.CALCulate(Ch).SELected.ZPARameter.DEFine

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y[:SCALe]:ABSolute:UNIT {String}

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y[:SCALe]:ABSolute:UNIT?

Query Response

{String}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:ABS:UNIT Sample_Text"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:ABS:UNIT?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.AUTO

Object type

Method (**Write Only**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.AUTO

Description

This command executes the auto scale function, for the selected trace (*Tr*) of the selected channel (*Ch*). The Auto Scale function automatically adjusts the value of the reference division line and the scale per division to display the trace appropriately.

Examples

SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.AUTO

Related objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel

Equivalent key

Scale > **Auto Scale**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y[:SCALe]:AUTO

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:AUTO"
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.BOTTom

Object Type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.BOTTom = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.BOTTom

Description

This command sets or gets the minimum scale value for the Log-Y Axis.

Variable

Parameter	*Value*
Description	Minimum scale value for Log-Y-Axis
Data Type	Double precision floating point type (Double)
Range	1a ~ 500P
Preset Value	1m
Unit	Depending on Measurement format
Resolution	-

Examples

See SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SPACing

Related Objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SPACing

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.TOP

Equivalent Key

Scale > **Log Y-Axis Top/Bottom** > **Bottom Value**

Using either the keyboard or ENTRY keys on the front panel, enter the bottom value.

Equivalent SCPI Command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y[:SCALe]:BOTTom <numeric>

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y[:SCALe]:BOTTom?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:BOTT 0.000000000001"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:BOTT?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.PDIVision

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.PDIVision = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.PDIVision

Description

For the selected trace (*Tr*) of selected channel (*Ch*), when the data format is not the Smith chart format or the polar format, sets the scale per division. When the data format is the Smith chart format or the polar format, sets the full scale value (the value of the outermost circumference).

Variable

Parameter	*Value*
Description	Scale value
Data type	Double precision floating point type (Double)
Range	1E-18 to 1E8
Preset value	Varies depending the data format. • Log magnitude: 10 • Phase, Expanded phase or Positive phase: 90 • Group delay: 1E-8 • Smith chart or Polar or SWR: 1 • Linear magnitude: 0.1 • Real or Imaginary: 0.2
Unit	Varies depending on the data format. • Log magnitude: dB (decibel) • Phase, Expanded phase or Positive phase: ° (degree) • Group delay: s (second) • Others: No unit

Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Pdiv As Double
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.FORMat = "gdel"
SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.PDIVision = 1E-9
Pdiv = SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.PDIVision
```

Related objects

SCPI.CALCulate(Ch).SELected.FORMat

SCPI.DISPlay.WINDow(Ch).Y.SCALe.DIVisions

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RPOSition

Equivalent key

Scale > **Scale/Div**

Equivalent SCPI command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :Y[:SCALe]:PDIVision <numeric>

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :Y[:SCALe]:PDIVision?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:PDIV 2.5"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:PDIV?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RLEVel

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RLEVel = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RLEVel

Description

This command sets/gets the value of the reference division line, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Value of reference division line
Data type	Double precision floating point type (Double)
Range	-5E8 to 5E8
Preset value	0
Unit	Varies depending on the data format. • Log magnitude (MLOG): dB (decibel) • Phase (PHAS), Expanded phase (UPH) or Positive phase (PPH): ° (degree) • Group delay (GDEL): s (second) • Others: No unit
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim RefLvl As Double
SCPI.CALCulate(1).PARameter(2).SELect
SCPI.CALCulate(1).SELected.FORMat = "phas"
```

```
SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RLEVel = 90
Pdiv = SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RLEVel
```

Related objects

SCPI.CALCulate(Ch).SELected.FORMat

SCPI.DISPlay.WINDow(Ch).Y.SCALe.DIVisions

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RPOSition

Equivalent key

Scale > **Reference Value**

Equivalent SCPI command

Syntax

```
:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :Y[:SCALe]:RLEVel <numeric>
:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :Y[:SCALe]:RLEVel?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RLEV 1E2"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RLEV?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RPOSition

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*T36r*).Y.SCALe.RPOSition = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RPOSition

Description

This command specifies the position of a reference division line with its number (an integer assigned starting from 0 from the lowest division), for the selected trace (*Tr*) of selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Position of reference division line
Data type	Long integer type (Long)
Range	0 to the number of divisions
Preset value	5
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim RefPos As Long
SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RPOSition = 6
RefPos = SCPI.DISPlay.WINDow(1).TRACe(2).Y.SCALe.RPOSition
```

Related objects

SCPI.CALCulate(Ch).SELected.FORMat

SCPI.DISPlay.WINDow(Ch).Y.SCALe.DIVisions

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel

Equivalent key

Scale > **Reference Position**

Equivalent SCPI command

Syntax

```
:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :Y[:SCALe]:RPOSition
<numeric>
:DISPlay:WINDow{[1]-4}:TRACe{[1]-4} :Y[:SCALe]:RPOSition?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RPOS 6"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:RPOS?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.TOP

Object Type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.TOP = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SCALe.TOP

Description

This command sets or gets the maximum scale value for the Log-Y Axis.

Variable

Parameter	*Value*
Description	Maximum scale value for Log-Y-Axis
Data Type	Double precision floating point type (Double)
Range	2a ~ 1E
Preset Value	1k
Unit	Depending on Measurement format
Resolution	-

Examples

See SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SPACing

Related Objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SPACing

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.BOTTom

Equivalent Key

Scale > **Log Y-Axis Top/Bottom** > **Top Value**

Using either the keyboard or ENTRY keys on the front panel, enter the top value.

Equivalent SCPI Command

Sntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y[:SCALe]:TOP <numeric>

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y[:SCALe]:TOP?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:TOP 0.000000000002"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:TOP?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SPACing

Object Type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SPACing = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(Tr).Y.SPACing

Description

This command sets or gets Y-Axis format in Log or Linear format.

Variable

Parameter	*Param*
Description	Vertical axis display type of the graph (in Log or Linear)
Data Type	Character string type (String)
Range	LINear|LOGarithmic Select from the following: "LINear": Sets Y-Axis to Linear scale. "LOGarithmic" : Sets Y-Axis to Log scale.
Preset Value	LINear
Unit	-
Resolution	-

Examples

```
Dim Bottom as Double, Top As Double


SCPI.DISPlay.WINDow(1).TRACe(1).Y.SPACing = "LOG"

SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.BOTTom = 0.00000001
SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.TOP =1000


Bottom = SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.BOTTom
Top = SCPI.DISPlay.WINDow(1).TRACe(1).Y.SCALe.TOP
```

Related Objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.BOTTom

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.TOP

Equivalent Key

Scale > **Y-Axis**

Equivalent SCPI Command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y:SPACing {LIN|LOG}

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y:SPACing?

Query Response

{LIN|LOG}<newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC1:Y:SPAC LIN"
20 OUTPUT 717;":DISP:WIND1:TRAC1:Y:SPAC?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.TRACk.FREQuency

Object Type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.TRACk.FREQuency = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.TRACk.FREQuency

Description

This command sets/gets a frequency when you want to specify a frequency on the trace data as the reference value, for the selected trace (*Tr*) of the selected channel (*Ch*).

NOTE Tracking is not performed when the specified frequency lies outside the preset range. When a frequency that does not match any measurement point is specified, interpolation is performed using the preceding and following measurement points, and the resulting value is used as the reference value for tracking.

Variable

Parameter	*Value*
Description	Frequency for tracking
Data Type	Double precision floating point type (Double)
Range	-1E+12 to 1E+12
Preset Value	0
Unit	Hz (hertz)

Examples

```
Dim RefFreq as Double
SCPI.DISPlay.WINDow(1).TRACe(4).Y.TRACk.FREQuency = 945E8
RefFreq = SCPI.DISPlay.WINDow(1).TRACe(4).Y.TRACk.FREQuency
```

Related Objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.TRACk.MODE

Equivalent Key

Scale > **Reference Tracking** > **Track Frequency**

Equivalent SCPI Command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y:TRACk:FREQuency <numeric>

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y:TRACk:FREQuency?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC2:Y:TRAC:FREQ 1.3E9"
20 OUTPUT 717;":DISP:WIND1:TRAC2:Y:TRAC:FREQ?"
30 ENTER 717;A
```

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.TRACk.MODE

Object Type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.TRACk.MODE = *Param*

Param = SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.TRACk.MODE

Description

This command sets/gets the tracking method to offset the trace data after sweep, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Tracking method to offset the trace data
Data Type	Character string type (String)
Range	Select from the following: • "OFF" : Does not perform tracking for trace data. • "PEAK": Specifies the peak value as the reference value. • "FREQuency": Specifies the specified frequency as the reference value.
Preset Value	"OFF"

Examples

```
Dim TracMode as String
SCPI.DISPlay.WINDow(2).TRACe(1).Y.TRACk.MODE = "FREQ"
TracMode = SCPI.DISPlay.WINDow(2).TRACe(1).Y.TRACk.MODE
```

Related Objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.TRACk.FREQuency

Equivalent Key

Scale > **Reference Tracking** > **Tracking** > **Off** | **Track Peak** | **Track Freq**

Equivalent SCPI Command

Syntax

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y:TRACk:MODE {OFF|PEAK|FREQuency}

:DISPlay:WINDow{[1]-4}:TRACe{[1]-4}:Y:TRACk:MODE?

Query Response

{OFF|PEAK|FREQ} <newline><^END>

Example of use

```
10 OUTPUT 717;":DISP:WIND1:TRAC4:Y:TRAC:MODE FREQ"
20 OUTPUT 717;":DISP:WIND1:TRAC4:Y:TRAC:MODE?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).X.SPACing

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).X.SPACing = *Param*

Param = SCPI.DISPlay.WINDow(*Ch*).X.SPACing

Description

This command selects the display type of the graph horizontal axis of the selected channel (*Ch*) for segment sweep.

Variable

Parameter	*Param*
Description	Horizontal axis display type of the graph for segment sweep
Data type	Character string type (String)
Range	Select from the following: • "LINear": Specifies the frequency base (linear frequency axis with the minimum frequency at the left edge and the maximum frequency at the right edge). • "OBASe" : Specifies the order base (axis in which the measurement point numbers are positioned evenly in the order of measurement).
Preset value	"OBASe"

Examples

```
Dim DispSegm As String
SCPI.SENSe(1).SWEep.TYPE = "segm"
SCPI.DISPlay.WINDow(1).X.SPACing = "obas"
DispSegm = SCPI.DISPlay.WINDow(1).X.SPACing
```

Related objects

SCPI.SENSe(Ch).SWEep.TYPE

Equivalent key

Sweep Setup > **Segment Display** > **Freq Base**|**Order Base**

Equivalent SCPI command

Syntax

```
:DISPlay:WINDow{[1]-4}:X:SPACing {LINear|OBASe}
:DISPlay:WINDow{[1]-4}:X:SPACing?
```

Query response

```
{LIN|OBAS}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":DISP:WIND1:X:SPAC OBAS"
20 OUTPUT 717;":DISP:WIND1:X:SPAC?"
30 ENTER 717;A$
```

SCPI.DISPlay.WINDow(*Ch*).Y.SCALe.DIVisions

Object type

Property (**Read-Write**)

Syntax

SCPI.DISPlay.WINDow(*Ch*).Y.SCALe.DIVisions = *Value*

Value = SCPI.DISPlay.WINDow(*Ch*).Y.SCALe.DIVisions

Description

This command sets/gets the number of divisions in all the graphs, for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of divisions of graph
Data type	Long integer type (Long)
Range	4 to 30
Preset value	10
Resolution	2
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Divs As Long
SCPI.DISPlay.WINDow(1).Y.SCALe.DIVisions = 12
Divs = SCPI.DISPlay.WINDow(1).Y.SCALe.DIVisions
```

Related objects

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.PDIVision

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RLEVel

SCPI.DISPlay.WINDow(Ch).TRACe(Tr).Y.SCALe.RPOSition

Equivalent key

Scale > **Divisions**

Equivalent SCPI command

Syntax

```
:DISPlay:WINDow{[1]-4}:Y[:SCALe]:DIVisions <numeric>
:DISPlay:WINDow{[1]-4}:Y[:SCALe]:DIVisions?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":DISP:WIND1:Y:DIV 12"
20 OUTPUT 717;":DISP:WIND1:Y:DIV?"
30 ENTER 717;A
```

FORMAT

SCPI.FORMat.BORDer

Object type

Property (**Read-Write**)

Syntax

SCPI.FORMat.BORDer = *Param*

Param = SCPI.FORMat.BORDer

Description

This command sets/gets the transfer order of each byte in the output data (byte order), when the data transfer format is set to binary mode (by specifying "REAL" with SCPI.FORMat.DATA object).

NOTE This object is NOT used when controlling the E5061B using COM objects through E5061B VBA.

Variable

Parameter	*Param*
Description	Byte order
Data type	Character string type (String)
Range	Select from the following: "NORMal": Specifies the byte order in which transfer starts from the byte including MSB (Most Significant Bit). "SWAPped": Specifies the byte order in which transfer starts from the byte including LSB (Least Significant Bit).
Preset value	"NORMal"

Examples

```
Dim BitOrd As String
SCPI.FORMat.BORDer = "swap"
BitOrd = SCPI.FORMat.BORDer
```

Related objects

SCPI.FORMat.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

```
:FORMat:BORDer {NORMal|SWAPped}
:FORMat:BORDer?
```

Query response

```
{NORM|SWAP}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":FORM:BORD SWAP"
20 OUTPUT 717;":FORM:BORD?"
30 ENTER 717;A$
```

SCPI.FORMat.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.FORMat.DATA = *Param*

Param = SCPI.FORMat.DATA

Description

This command can be used to set/get format data using the following SCPI commands:

SCPI.CALCulate(Ch).SELected.DATA.FDATa

SCPI.CALCulate(Ch).SELected.DATA.FMEMory

SCPI.CALCulate(Ch).SELected.DATA.SDATa

SCPI.CALCulate(Ch).SELected.DATA.SMEMory

SCPI.CALCulate(Ch).SELected.FUNCtion.DATA

SCPI.CALCulate(Ch).SELected.LIMit.DATA

SCPI.CALCulate(Ch).SELected.LIMit.REPort.ALL

SCPI.CALCulate(Ch).SELected.LIMit.REPort.DATA

SCPI.CALCulate(Ch).SELected.BLIMit.REPort.DATA

SCPI.CALCulate(Ch).SELected.RLIMit.DATA

SCPI.CALCulate(Ch).SELected.RLIMit.REPort.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.GPData

SCPI.SENSe(Ch).FREQuency.DATA

SCPI.SENSe(Ch).SEGMent.DATA

NOTE ASCII transfer format must be specified when controlling the E5061B using SCPI commands with the Parse object in the E5061B VBA.

Variable

Parameter	*Param*
Description	Data transfer format
Data type	Character string type (String)

Range	Select from the following: • "ASCii": Specifies the ASCII transfer format. • "REAL": Specifies the IEEE 64-bit floating point binary transfer format. • "REAL32": Specifies the IEEE 32-bit floating point binary transfer format.
Preset value	"ASCii"

Examples

```
Dim Fmt As String
SCPI.FORMat.DATA = "ASC"
Fmt = SCPI.FORMat.DATA
```

Related objects

SCPI.FORMat.BORDer

Parse

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

```
:FORMat:DATA {ASCii|REAL|REAL32}
:FORMat:DATA?
```

Query response

```
{ASC|REAL|REAL32}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":FORM:DATA REAL"
20 OUTPUT 717;":FORM:DATA?"
30 ENTER 717;A$
```

SCPI.FORMat.REAL.ASCii.LENGth

Object Type

Property (**Read-Write**)

Syntax

SCPI.FORMat.REAL.ASCii.LENGth = *Value*

Value = SCPI.FORMat.REAL.ASCii.LENGth

Description

This command sets and gets the number of significant digits of a floating point number format in Ascii format. See ASCII Transfer Format.

For example, 1000 is expressed as :

- When 12 is set, "+1.00000000000E+003."
- When 14 is set, "+1.0000000000000E+003."

Variable

Parameter	*Value*
Description	Number of significant digits
Data Type	Long integer type (Long)
Range	12|14
Preset Value	12
Unit	-
Resolution	-

Examples

```
Dim Var as Long
SCPI.FORMat.REAL.ASCii.LENGth = 14
Var = SCPI.FORMat.REAL.ASCii.LENGth
```

Related Objects

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:FORMat:REAL:ASCii:LENGth {12|14}

:FORMat:REAL:ASCii:LENGth?

Query Response

{12|14}<newline><^END>

Example of use

```
10 OUTPUT 717;":FORM:REAL:ASC:LENG 14"
20 OUTPUT 717;":FORM:REAL:ASC:LENG?"
30 ENTER 717;A
```

HCOPY

SCPI.HCOPy.ABORt

Object type

Method (**Write Only**)

Syntax

SCPI.HCOPy.ABORt

Description

This command aborts the print output.

Examples

SCPI.HCOPy.ABORt

Related objects

SCPI.HCOPy.IMMediate

Equivalent key

System > **Abort Printing**

Equivalent SCPI command

Syntax

:HCOPy:ABORt

Example of use

```
10 OUTPUT 717;":HCOP:ABOR"
```

SCPI.HCOPy.SDUMp.DATA.FORMat

Object Type

Property (**Read-Write**)

Syntax

SCPI.HCOPy.SDUMp.DATA.FORMat = *Value*

Value = SCPI.HCOPy.SDUMp.DATA.FORMat

Description

This command sets or gets the image format (print screen image).

Variable

Parameter	*Value*
Description	Image format
Data Type	Character string type (String)
Range	PNG|BMP
Preset Value	PNG
Unit	-
Resolution	-

Examples

```
Dim PixType as String
PixType = "PNG"
SCPI.HCOPy.SDUMp.DATA.FORMat = PixType

ReadPixType = SCPI.HCOPy.SDUMp.DATA.FORMat
```

Related Objects

SCPI.HCOPy.SDUMp.DATA.IMMediate

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:HCOPy:SDUMp:DATA:FORMat {PNG|BMP}

:HCOPy:SDUMp:DATA:FORMat?

Query Response

{PNG|BMP}<newline><^END>

Example of use

```
10 OUTPUT 717;":HCOP:SDUM:DATA:FORM PNG"
20 OUTPUT 717;":HCOP:SDUM:DATA:FORM?"
30 ENTER 717;A$
```

SCPI.HCOPy.SDUMp.DATA.IMMediate

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.HCOPy.SDUMp.DATA.IMMediate

Description

This command gets print screen image.

Variable

Parameter	*Value*
Description	print screen
Data Type	Binary (Byte)
Range	-
Preset Value	-
Unit	-
Resolution	-

Examples

```
Dim TmpData() as Byte
Open "C:\Screen.PNG" For Binary As #1
TmpData() = SCPI.HCOPy.SDUMp.DATA.IMMediate
Put #1, , TmpData()
Close #1
```

Related Objects

SCPI.HCOPy.SDUMp.DATA.FORMat

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:HCOPy:SDUMp:DATA[:IMMediate]?

Query Response

<Binary><newline><^END>

Example of use (VISA-COM)

```
'*** Save the screen image to the file
Age506x.WriteString ":HCOP:SDUM:DATA:FORM PNG", True
Age506x.WriteString ":HCOP:SDUM:DATA?", True
PNGdata = Age506x.ReadIEEEBlock(BinaryType_UI1, False, True)

Open "C:\TEST.png" For Binary As #1
   Put #1, , PNGdata()
Close
```

SCPI.HCOPy.IMAGe

Object type

Property (**Read-Write**)

Syntax

SCPI.HCOPy.IMAGe = *Param*

Param = SCPI.HCOPy.IMAGe

Description

This command sets/gets the print color for output (to the printer).

Variable

Parameter	*Param*
Description	Print color for output to the printer.
Data type	Character string type (String)
Range	Select from the following: • "NORMal": Specifies printing in close color to the display color. • "INVert": Specifies printing in the inverted color of the display color.
Preset value	"INVert"

Examples

```
Dim Img As String
SCPI.HCOPy.IMAGe = "norm"
Img = SCPI.HCOPy.IMAGe
```

Related objects

SCPI.HCOPy.IMMediate

Equivalent key

System > **Invert Image**

Equivalent SCPI command

Syntax

:HCOPy:IMAGe {NORMal|INVert}

:HCOPy:IMAGe?

Query response

{NORM|INV}<newline><^END>

Example of use

```
10 OUTPUT 717;":HCOP:IMAG NORM"
20 OUTPUT 717;":HCOP:IMAG?"
30 ENTER 717;A$
```

SCPI.HCOPy.IMMediate

Object type

Method (**Write Only**)

Syntax

SCPI.HCOPy.IMMediate

Description

This command outputs the display image on the LCD display to the printer connected to the E5061B.

NOTE When printing the E5061B measurement screen, execute the VBA program with the Visual Basic application closed. For the method, see Running a Program from the E5061B Measurement Screen.

Examples

SCPI.HCOPy.IMMediate

Related objects

SCPI.HCOPy.ABORt

SCPI.HCOPy.IMAGe

Equivalent key

System > **Print**

When performing the operation from the front panel, the image on the LCD display memorized in the volatile memory (clipboard) (the image on the LCD display when the **Capture** (**System**) key is pressed) is printed. If no image is memorized in the clipboard, in the same way as the SCPI.HCOPy.IMMediate object, the image on the LCD display at the execution is memorized in the clipboard and then it is printed.

Equivalent SCPI command

Syntax

:HCOPy[:IMMediate]

Example of use

```
10 OUTPUT 717;":HCOP"
```

IEEE

SCPI.IEEE4882.CLS

Object type

Method (**Write Only**)

Syntax

SCPI.IEEE4882.CLS

Description

This command clears the following:

- Error Queue
- Status Byte Register
- Standard Event Status Register
- Operation Status Event Register
- Questionable Status Event Register
- Questionable Limit Status Event Register
- Questionable Limit Channel Status Event Register
- Questionable Bandwidth Limit Status Event Register
- Questionable Bandwidth Limit Channel Status Event Register
- Questionable Ripple Limit Status Event Register
- Questionable Ripple Limit Channel Status Event Register

Examples

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*CLS

Example of use

```
10 OUTPUT 717;"*CLS"
```

SCPI.IEEE4882.ESE

Object type

Property (**Read-Write**)

Syntax

SCPI.IEEE4882.ESE = *Value*

Value = SCPI.IEEE4882.ESE

Description

This command sets/gets the value of the Standard Event Status Enable Register.

Variable

Parameter	*Value*
Description	Value of the Standard Event Status Enable Register
Data type	Long integer type (Long)
Range	0 to 255
Preset value	0
Note	If the specified variable is out of the allowable setup range, the result of bitwise AND with 255 (0xff) is set.

Examples

```
Dim Stat As Long
SCPI.IEEE4882.ESE = 16
Stat = SCPI.IEEE4882.ESE
```

Related objects

SCPI.IEEE4882.SRE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*ESE <numeric>

*ESE?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;"*ESE 16"
20 OUTPUT 717;"*ESE?"
30 ENTER 717;A
```

SCPI.IEEE4882.ESR

Object type

Property (**Read Only**)

Syntax

Value = SCPI.IEEE4882.ESR

Description

This command reads the value of the Standard Event Status Register. Execution of this command clears the register value.

Variable

Parameter	*Value*
Description	Value of the Standard Event Status Register
Data type	Long integer type (Long)
Preset value	128

Examples

```
Dim Stat As Long
Stat = SCPI.IEEE4882.ESR
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*ESR?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;"*ESR?"
20 ENTER 717;A
```

SCPI.IEEE4882.IDN

Object type

Property (**Read Only**)

Syntax

Cont = SCPI.IEEE4882.IDN

Description

This command reads the product information (manufacturer, model number, serial number, and firmware revision number) of the E5061B.

Variable

Parameter	*Cont*
Description	Product information ("{string 1},{string 2},{string 3},{string 4}") • {string 1}: Manufacturer. Agilent Technologies is always read out. • {string 2}: Model number. E5061B is always read out. • {string 3}: Serial number (example: MY123400101). • {string 4}: Firmware revision number (example: A.07.00).
Data type	Character string type (String)

Examples

```
Dim Who As String
Who = SCPI.IEEE4882.IDN
```

Equivalent key

System > **Firmware Revision**

Equivalent SCPI command

Syntax

*IDN?

Query response

{string 1},{string 2},{string 3},{string 4}<newline><^END>

Example of use

```
10 OUTPUT 717;"*IDN?"
20 ENTER 717;A$
```

SCPI.IEEE4882.LRN

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.IEEE4882.LRN

Description

This command gets the device setup query. This command is defined as "Learn Device Setup Query" in IEEE 488.2. The command returns instrument settings by binary block data (same as Save/Recall state file contents) with "SYSTem:SET " prefix.

The returned data is the same as the contents of state file which can be saved by the SCPI.MMEMory.STORe.STATe. Therefore, the returned data contents is changed according to the setting of SCPI.MMEMory.STORe.STYPe.

Variable

Parameter	*Value*
Description	Setting Data
Data Type	Binary Type (Byte)
Range	-
Preset Value	-
Unit	-
Resolution	-

Examples

```
'To write LRN data
Dim TempWrite() As Byte
Open "C:\LRN.dat" For Binary As #1
TempWrite() = SCPI.IEEE4882.LRN
Put #1, , TempWrite()
Close #1
```

To save/get the E5061B state for COM, use SCPI.MMEMory.STORe.STATe and SCPI.MMEMory.LOAD.STATe.

Related objects

SCPI.SYSTem.SET

SCPI.MMEMory.LOAD.STATe

SCPI.MMEMory.STORe.STATe

SCPI.MMEMory.STORe.STYPe

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

*LRN?

Query Response

<binary><newline><^END>

Example of use (VISA-COM)

```
Dim LRNData() As Byte, SETData() As Byte, NoofByte As Double
 '*** Get the LRN data as Binaray data
 Age506x.WriteString "*LRN?", True
 LRNData = Age506x.ReadIEEEBlock(BinaryType_UI1, False, True)

 '*** Save the LRN data in the file
Open "C:\LRN.dat" For Binary As #1
        Put #1, , LRNData()
Close

MsgBox "Get Data"

'*** Recall the LRN data from the file
Open "C:\LRN.dat" For Binary As #1
        NoofByte = LOF(1)
       ReDim SETData(NoofByte)
       Get #1, , SETData()
Close
```

```
'*** Send the LRN data to E5061B
Age506x.IO.Write SETData, NoofByte
```

SCPI.IEEE4882.OPC

Object type

Property (**Read-Write**)

Syntax

(1)SCPI.IEEE4882.OPC = *Dummy*

(2)*Value* = SCPI.IEEE4882.OPC

Description

Case (1):

This command sets/gets 1 the OPC bit (bit 0) of the Standard Event Status Register when all of pending operations complete. For information on the structure of the status register, see Status Reporting System.

Variable

Case (2):

Parameter	*Value*
Description	1 is returned when all pending operations are completed
Data type	Long integer type (Long)

Examples

```
Dim Opcbit as Long
SCPI.IEEE4882.OPC = 1
Opcbit = SCPI.IEEE4882.OPC
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.ISOLation

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.ACQuire.THRU

SCPI.TRIGger.SEQuence.SINGle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*OPC

Example of use

```
10 OUTPUT 717;"*OPC"
```

SCPI.IEEE4882.OPT

Object type

Property (**Read Only**)

Syntax

Cont = SCPI.IEEE4882.OPT

Description

This command reads the identification numbers of options installed in the E5061B.

Variable

Parameter	*Cont*
Description	Identification numbers of installed options
Data type	Character string type (String)
Note	If there is no installed option, 0 is read out.

Examples

```
Dim OptNum As String
OptNum = SCPI.IEEE4882.OPT
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*OPT?

Query response

{numeric}<newline><^END>

Example of use

```
Call viVPrintf(vi, "*OPT?" + vbLf, 0)
Call viVScanf(vi, "%t", Result)
```

SCPI.IEEE4882.RST

Object type

Method (**Write Only**)

Syntax

SCPI.IEEE4882.RST

Description

This command presets the E5061B to its default settings and is different from setting state preset with the SCPI.SYSTem.PRESet object as the continuous initiation mode (see SCPI.INITiate(Ch).CONTinuous) of channel 1 is set to OFF.

Examples

SCPI.IEEE4882.RST

Related objects

SCPI.INITiate(Ch).CONTinuous

SCPI.SYSTem.PRESet

SCPI.SYSTem.UPReset

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*RST

Example of use

```
10 OUTPUT 717;"*RST"
```

SCPI.IEEE4882.SRE

Object type

Property (**Read-Write**)

Syntax

SCPI.IEEE4882.SRE = *Value*

Value = SCPI.IEEE4882.SRE

Description

This command sets/gets the value of Service Request Enable Register.

Variable

Parameter	*Value*
Description	Value of the Service Request Enable Register
Data type	Long integer type (Long)
Range	0 to 255
Preset value	0
Note	If the specified variable is out of the allowable setup range, the result of bitwise AND with 255 (0xff) is set. Note that bit 6 cannot be set to 1.

Examples

```
Dim Stat As Long
SCPI.IEEE4882.SRE = 8
Stat = SCPI.IEEE4882.SRE
```

Related objects

SCPI.IEEE4882.ESE

SCPI.STATus.OPERation.ENABle

SCPI.STATus.QUEStionable.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*SRE <numeric>

*SRE?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;"*SRE 128"
20 OUTPUT 717;"*SRE?"
30 ENTER 717;A
```

SCPI.IEEE4882.STB

Object type

Property (**Read Only**)

Syntax

Value = SCPI.IEEE4882.STB

Description

This command reads the value of Status Byte Register.

Variable

Parameter	*Value*
Description	Value of the Status Byte Register
Data type	Long integer type (Long)
Preset Value	0

Examples

```
Dim Stat As Long
Stat = SCPI.IEEE4882.STB
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*STB?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;"*STB?"
20 ENTER 717;A
```

SCPI.IEEE4882.TRG

Object type

Method (**Write Only**)

Syntax

SCPI.IEEE4882.TRG

Description

This command triggers the E5061B if the trigger source is set to GPIB/LAN (set to BUS with the SCPI.TRIGger.SEQuence.SOURce).

Examples

```
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.IEEE4882.TRG
```

Related objects

SCPI.TRIGger.SEQuence.SOURce

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*TRG

Example of use

```
10 OUTPUT 717;"*TRG"
```

SCPI.IEEE4882.TST

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.IEEE4882.TST

Description

Does nothing. The self-test is not executed by this command in the case of the E5061B. Always returns 0.

Variable

Parameter	*Value*
Description	Self-test query
Data Type	Long integer type (Long)
Range	0 (Always returns 0)
Preset Value	0

Examples

```
Dim Dmy as Long
Dmy = SCPI.IEEE4882.TST
```

Related Objects

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

*TST?

Query Response

{numeric} <newline><^END>

NOTE Always returns 0.

Example of use

```
10 OUTPUT 717;"*TST?"
20 ENTER 717;A
```

SCPI.IEEE4882.WAI

Object type

Method (**Write Only**)

Syntax

SCPI.IEEE4882.WAI

Description

This command waits for the execution of all objects sent before this command is completed.

Examples

```
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.TRIGger.SEQuence.SINGle
SCPI.IEEE4882.WAI
MsgBox "Done"
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

*WAI

Example of use

```
10 OUTPUT 717;"*WAI"
```

INIT

SCPI.INITiate(*Ch*).CONTinuous

Object type

Property (**Read-Write**)

Syntax

SCPI.INITiate(*Ch*).CONTinuous = *Status*

Status = SCPI.INITiate(*Ch*).CONTinuous

Description

This command turns ON/OFF the continuous initiation mode (setting by which the trigger system initiates continuously) of the selected channel (*Ch*) in the trigger system. For more information on the trigger system, see Section Trigger System.

Variable

Parameter	*Status*
Description	ON/OFF of the continuous initiation mode
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the continuous initiation mode. • False or OFF: Turns OFF the continuous initiation mode.
Preset value	Varies depending on [variable (*Ch*)]

Examples

```
Dim ContMode As Boolean
SCPI.INITiate(2).CONTinuous = True
ContMode = SCPI.INITiate(2).CONTinuous
```

Related objects

SCPI.INITiate(Ch).IMMediate

Equivalent key

Trigger > **Continuous** (ON)

Trigger > **Hold** (OFF)

Trigger > **Hold All Channels** (OFF for all channels)

Trigger > **Continuous Disp Channels** (ON for displayed channels)

Equivalent SCPI command

Syntax

:INITiate{[1]-4}:CONTinuous {ON|OFF|1|0}

:INITiate{[1]-4}:CONTinuous?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":INIT1:CONT OFF"
20 OUTPUT 717;":INIT1:CONT?"
30 ENTER 717;A
```

SCPI.INITiate(*Ch*).IMMediate

Object type

Method (**Write Only**)

Syntax

SCPI.INITiate(*Ch*).IMMediate

Description

This command changes the state of each channel of the selected channel (*Ch*) to the initiation state in the trigger system.

When this object is executed for a channel in the idle state in the trigger system, it goes into the initiation state immediately. Then, after measurement is executed once, it goes back to the idle state.

If this object is executed for a channel that is not in the idle state or a channel for which the continuous initiation mode is set to ON (setting by which the trigger system initiates continuously) in the trigger system, an error occurs when executed and the object is ignored. For more information on the trigger system, see Trigger System.

Examples

```
SCPI.INITiate(1).CONTinuous = False
SCPI.INITiate(1).IMMediate
```

Related objects

SCPI.INITiate(Ch).CONTinuous

Equivalent key

Trigger > **Single**

Equivalent SCPI command

Syntax

:INITiate{[1]-4}[:IMMediate]

Example of use

```
10 OUTPUT 717;":INIT1"
```

INPUT

SCPI.INPut.ATTenuation.GPPort.R

Object Type

Property (**Read-Write**)

Syntax

SCPI.INPut.ATTenuation.GPPort.R = *Value*

Value = SCPI.INPut.ATTenuation.GPPort.R

Description

This command sets/gets the input attenuation of port R for gain phase measurement.

Variable

Parameter	*Value*
Description	input attenuation of port R
Data Type	Double precision floating point type (Double)
Range	0 or 20
Preset Value	20
Unit	dB

Examples

```
Dim AttPortR as Double
SCPI.INPut.ATTenuation.GPPort.R = 20
AttPortR = SCPI.INPut.ATTenuation.GPPort.R
```

Related Objects

SCPI.INPut.ATTenuation.GPPort.T

SCPI.INPut.IMPedance.GPPort.R

SCPI.INPut.IMPedance.GPPort.T

Equivalent Key

Meas > **Gain-Phase** > **R Attenuator** > **0dB** | **20dB**

System > **Overload Recovery** > **R Attenuator** > **0dB** | **20dB**

Equivalent SCPI Command

Syntax

:INPut:ATTenuation:GPPort:R <numeric>

:INPut:ATTenuation:GPPort:R?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":INP:ATT:GPP:R 20"
20 OUTPUT 717;":INP:ATT:GPP:R ?"
30 ENTER 717;A
```

SCPI.INPut.ATTenuation.GPPort.T

Object Type

Property (**Read-Write**)

Syntax

SCPI.INPut.ATTenuation.GPPort.T = *Value*

Value = SCPI.INPut.ATTenuation.GPPort.T

Description

This command sets/gets the input attenuation of port T for gain phase measurement.

Variable

Parameter	*Value*
Description	input attenuation of port T
Data Type	Double precision floating point type (Double)
Range	0 or 20
Preset Value	20
Unit	dB

Examples

```
Dim AttPortT as Double
SCPI.INPut.ATTenuation.GPPort.T = 20
AttPortT = SCPI.INPut.ATTenuation.GPPort.T
```

Related Objects

SCPI.INPut.ATTenuation.GPPort.R

SCPI.INPut.IMPedance.GPPort.R

SCPI.INPut.IMPedance.GPPort.T

Equivalent Key

Meas > **Gain-Phase** > **T Attenuator** > **0dB** | **20dB**

System > **Overload Recovery** > **T Attenuator** > **0dB** | **20dB**

Equivalent SCPI Command

Syntax

:INPut:ATTenuation:GPPort:T <numeric>

:INPut:ATTenuation:GPPort:T?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":INP:ATT:GPP:T 20"
20 OUTPUT 717;":INP:ATT:GPP:T?"
30 ENTER 717;A
```

SCPI.INPut.IMPedance.GPPort.R

Object Type

Property (**Read-Write**)

Syntax

SCPI.INPut.IMPedance.GPPort.R = *Value*

Value = SCPI.INPut.IMPedance.GPPort.R

Description

This command sets/gets input impedance of port R for gain phase measurement.

Variable

Parameter	*Value*
Description	input impedance of port R
Data Type	Double precision floating point type (Double)
Range	50 or 1E+6
Preset Value	1E+6
Unit	Ω

Examples

```
Dim ImpPortR as Double
SCPI.INPut.IMPedance.GPPort.R = 50
ImpPortR = SCPI.INPut.IMPedance.GPPort.R
```

Related Objects

SCPI.INPut.ATTenuation.GPPort.R

SCPI.INPut.ATTenuation.GPPort.T

SCPI.INPut.IMPedance.GPPort.T

Equivalent Key

Meas > **Gain-Phase** > **R Input Z** > **50Ω** | **1MΩ**

System > **Overload Recovery** > **R Input Z** > **50Ω** | **1MΩ**

Equivalent SCPI Command

Syntax

:INPut:IMPedance:GPPort:R <numeric>

:INPut:IMPedance:GPPort:R?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":INP:IMP:GPP:R 1E+6"
20 OUTPUT 717;":INP:IMP:GPP:R?"
30 ENTER 717;A
```

SCPI.INPut.IMPedance.GPPort.T

Object Type

Property (**Read-Write**)

Syntax

SCPI.INPut.IMPedance.GPPort.T = *Value*

Value = SCPI.INPut.IMPedance.GPPort.T

Description

This command sets/gets input impedance of port T for gain phase measurement.

Variable

Parameter	*Value*
Description	input impedance of port T
Data Type	Double precision floating point type (Double)
Range	50 or 1E+6
Preset Value	1E+6
Unit	Ω

Examples

```
Dim ImpPortT as Double
SCPI.INPut.IMPedance.GPPort.T = 1E+6
ImpPortT = SCPI.INPut.IMPedance.GPPort.T
```

Related Objects

SCPI.INPut.ATTenuation.GPPort.R

SCPI.INPut.ATTenuation.GPPort.T

SCPI.INPut.IMPedance.GPPort.R

Equivalent Key

Meas > **Gain-Phase** > **T Input Z** > **50Ω** | **1MΩ**

System > **Overload Recovery** > **T Input Z** > **50Ω** | **1MΩ**

Equivalent SCPI Command

Syntax

:INPut:IMPedance:GPPort:T <numeric>

:INPut:IMPedance:GPPort:T?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":INP:IMP:GPP:T 50"
20 OUTPUT 717;":INP:IMP:GPP:T?"
30 ENTER 717;A
```

LXI

SCPI.LXI.IDENtify.STATe

Object Type

Property (**Read-Write**)

Syntax

SCPI.LXI.IDENtify.STATe = *Status*

Status = SCPI.LXI.IDENtify.STATe

Description

This command sets or gets the LXI Status Indicator state.

Variable

Parameter	*Status*
Description	ON/OFF LXI Control Identification
Data Type	Boolean type (Boolean)
Range	Select from the following: True or On: Sets the LXI Status Indicator to the 'IDENTIFY' state. False or Off: Set the LXI Status Indicator to the 'NORMAL' state.
Preset Value	False
Unit	-
Resolution	-

Examples

```
Dim Var as Boolean
SCPI.LXI.IDENtify.STATe = True
Var = SCPI.LXI.IDENtify.STATe
```

Equivalent Key

There is no equivalent key is available on the front panel. However, the similar key is:

System > **Misc Setup** > **Network Setup** > **LAN Dialog...**

Equivalent SCPI Command

Syntax

```
:LXI:IDENtify[:STATe] {ON|OFF|1|0}
:LXI:IDENtify[:STATe]?
```

Query Response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":LXI:IDEN ON"
20 OUTPUT 717;":LXI:IDEN?"
30 ENTER 717;A
```

MEMORY

:MMEMory:LOAD:PROGram

No equivalent COM Commands

Syntax

:MMEMory:LOAD:PROGram <string>

Description

This command loads (or imports) a VBA project (a file with the .vba extension), a module (a file with the .bas extension), a user form (a file with the .frm extension) or a class module (a file with the .cls extension). If the specified file does not exist, an error occurs and the command is ignored.

Variables

Parameter	*String*
Description	File name in which you want to load the VBA project
Range	254 characters or less
Preset value	""

Example of use

```
10 OUTPUT 717;":MMEM:LOAD:PROG ""Test1/Test1_01.vba"""
10 OUTPUT 717;":MMEM:LOAD:PROG ""A:Test1_01.vba"""
```

Related commands

:MMEM:STOR:PROG

Equivalent key

Macro Setup > **Load VBA Project**

:MMEMory:STORe:PROGram

No equivalent COM Commands

Syntax

:MMEMory:STORe:PROGram <string>

Description

This command saves a VBA project opened on the VBA editor into a file. The file name needs to have a .vba extension. If a file with the specified file name exists, its contents are overwritten.

Variables

Parameter	*String*
Description	File name in which you want to save the VBA project
Range	254 characters or less
Preset value	""

Example of use

```
10 OUTPUT 717;":MMEM:STOR:PROG ""Test1/Test1_01.vba"""
10 OUTPUT 717;":MMEM:STOR:PROG ""D:Test1_01.vba"""
```

Related commands

:MMEM:LOAD:PROG

Equivalent key

Macro Setup > **Save VBA Project**

:MMEMory:TRANsfer

No equivalent COM Commands

Syntax

:MMEMory:TRANsfer <string>,<block>

:MMEMory:TRANsfer? <string>

Description

This command sets/gets data to/from a file on the built-in storage device of the E5061B. By reading out data with this command and writing it to a file on the external controller, file transfer from the E5061B to the external controller can be realized.

On the other hand, by reading out data from the external controller and writing it to a file on the E5061B with this command, file transfer from the external controller to the E5061B can be realized.

When you use directory names and file names, separate them with "/" (slash) or "\" (backslash). If a file with the specified file name already exists for writing or if the specified file does not exist for reading out (Query), an error occurs and the command is ignored.

Variables

Parameter	*String*
Description	File name on the built-in storage device
Range	254 characters or less

Parameter	*block*
Description	Data written on/read out from the file
Range	GPIB: 20 Mbytes or less LAN: 100 Kbytes or less

Query response

{block}<newline><^END>

Example of use

```
10 OUTPUT 717;":MMEM:TRAN ""Trace01.csv"",#6012345";Dat$
```

```
10 OUTPUT 717;":MMEM:TRAN? ""Trace01.csv"""
20 ENTER 717 USING "#,A";A$
30 ENTER 717 USING "#,A";Digit$
40 Img$="#,"&Digit$&"A"
50 ENTER 717 USING Img$;Byte$
60 Img$=Byte$&"A"
70 ALLOCATE Dat$[VAL(Byte$)]
80 ENTER 717 USING Img$;Dat$
```

Equivalent key

No equivalent key is available on the front panel.

SCPI.MMEMory.CATalog(*Dir*)

Object type

Property (**Read Only**)

Syntax

Cont = SCPI.MMEMory.CATalog(*Dir*)

Description

This command reads the following information on the built-in storage device of the E5061B:

- Space in use
- Available space
- Name and size of all files (including directories) in the specified directory

NOTE To read out the information in the root directory (folder), specify "\ " (backslash). Separate between directory names (file name) with "\ " (back slash), or "/" (slash).

Variable

Parameter	*Cont*
Description	Directory information ("{A},{B},{Name 1},,{Size 1},{Name 2},,{Size 2}, ... ,{Name N},,{Size N}") Where N is the number of all files in the specified directory and n is an integer between 1 and N. {A}: Space in use of the built-in storage device (byte). {B}: Available space of the built-in storage device (byte). {Name n}: Name of the n-th file (directory). {Size n}: Size (byte) of the n-th file (directory). Always 0 for directories.
Data type	Character string type (String)

Parameter	*Dir*

Description	Directory name of which the information you want to read out
Data type	Character string type (String)
Range	254 characters or less

Examples

```
Dim DirCont As String
DirCont = SCPI.MMEMory.CATalog("d:\")
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:MMEMory:CATalog? <string 1>

Query response

{string 2}<newline><^END>

The format of the readout character string is as follows:

"{used_size},{free_size},{name 1},,{size 1}, ... ,{name N},,{size N}"

Where:

N is the number of all the files in the specified directory and n is an integer between 1 and N.

{used_size}:

Space in use of the built-in storage device (byte).

{free_size}:

Available space of the built-in storage device (byte).

{name n}:

Name of the n-th file (directory).

{size n}:

Size (byte) of the n-th file (directory). Always 0 for directories.

Example of use

```
10 DIM A$[1000]
20 OUTPUT 717;":MMEM:CAT? ""\"""
30 ENTER 717;A$
```

SCPI.MMEMory.COPY

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.COPY = *File*

Description

This command copies a file.

NOTE Specify the file name with the extension. When you use directory names (folder names) and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	Indicates 2 file names (copy source and copy destination). *File(0):* Copy source file name *File(1):* Copy destination file name The index of the array starts from 0.
Data type	Variant type (Variant)
Range	254 characters or less
Note	If the specified copy source file does not exist, an error occurs when executed and the object is ignored. Notice that, if a file with the same name as the specified copy destination file name exists, its contents are overwritten.

Examples

```
SCPI.MMEMory.COPY = Array("test/state01.sta","d:test01.sta")
```

```
Dim File(1) As Variant
File(0) = "test/state01.sta"
File(1) = "d:test01.sta"
SCPI.MMEMory.COPY = File
```

Equivalent key

Save/Recall > **Save State** > **File Dialog...**

Equivalent SCPI command

Syntax

:MMEMory:COPY <string 1>,<string 2>

Example of use

```
10 OUTPUT 717;":MMEM:COPY ""Test1/State01.sta"","d:Test1_01.sta"""
```

SCPI.MMEMory.DELete

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.DELete = *File*

Description

This command deletes an existing file or directory (folder).

NOTE When you delete a directory, all the files and directories in it are deleted.

Specify the file name with the extension.

When you specify a file (directory) under an existing directory, separate them with "\" (back slash), or "/" (slash).

To delete all the files in the directory (folder), specify "\" (backslash).

Variable

Parameter	*File*
Description	File name or directory name you want to delete
Data type	Character string type (String)
Range	254 characters or less
Note	If the specified file or directory does not exist, an error occurs when executed and the object is ignored.

Examples

SCPI.MMEMory.DELete = "d:\"

SCPI.MMEMory.DELete = "test/state01.sta"

Equivalent key

Save/Recall > **Save State** > **File Dialog...**

Equivalent SCPI command

Syntax

:MMEMory:DELete <string>

Example of use

```
10 OUTPUT 717;":MMEM:DEL ""Test1/State01.sta"""
10 OUTPUT 717;":MMEM:DEL ""D:State01.sta"""
```

SCPI.MMEMory.LOAD.CHANnel.COEFficient

Object Type

Method (**Write Only**)

Syntax

SCPI.MMEMory.LOAD.CHANnel.COEFficient = *Param*

Description

This command recalls the channel coefficient data of the active channel from the specified register.

Variable

Parameter	*Param*
Description	Register
Data Type	Character string type (String)
Range	Select from the following: • "A": Specifies register A. • "B": Specifies register B. • "C" :Specifies register C. • "D" :Specifies register D.
Note	If no channel state has been saved in the specified register, an error occurs and the object is ignored.

Examples

```
Dim Rgstr as String
Rgstr = "A"
SCPI.MMEMory.LOAD.CHANnel.COEFficient = Rgstr
```

Related Objects

SCPI.MMEMory.LOAD.CHANnel.STATe

SCPI.MMEMory.STORe.CHANnel.COEFficient

SCPI.MMEMory.STORe.CHANnel.STATe

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:MMEMory:LOAD:CHANnel:COEFficient {A|B|C|D}

Query Response

{A|B|C|D} <newline><^END>

Example of use

```
10 OUTPUT 717;":MMEM:LOAD:CHAN:COEF B"
```

SCPI.MMEMory.LOAD.CHANnel.STATe

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.LOAD.CHANnel.STATe = *Register*

Description

This command recalls the instrument state for an individual channel (saved with the SCPI.MMEMory.STORe.CHANnel.STATe object) from the specified register as the setting of the active channel.

Variable

Parameter	*Register*
Description	Register
Data type	Character string type (String)
Range	Select from the following: "A": Specifies register A. "B": Specifies register B. "C" :Specifies register C. "D" :Specifies register D.
Note	If no instrument state has been saved in the specified register, an error occurs and the object is ignored.

Examples

SCPI.MMEMory.LOAD.CHANnel.STATe = "a"

Related objects

SCPI.MMEMory.STORe.CHANnel.STATe

SCPI.DISPlay.WINDow(Ch).ACTivate

Equivalent key

Save/Recall > **Recall Channel** > **A**|**B**|**C**|**D**

Equivalent SCPI command

Syntax

:MMEMory:LOAD:CHANnel[:STATe] {A|B|C|D}

Example of use

```
10 OUTPUT 717;":MMEM:LOAD:CHAN A"
```

SCPI.MMEMory.LOAD.LIMit

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.LOAD.LIMit = *File*

Description

This command recalls the specified limit table file (file with the .csv extension saved with SCPI.MMEMory.STORe.LIMit), from the limit table for the active trace of the active channel.

NOTE Specify the file name with the extension. When you use directory names and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name of limit table (extension ".csv")
Data type	Character string type (String)
Range	254 characters or less
Note	If the specified file does not exist, an error occurs when executed and the object is ignored.

Examples

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.LIMit = "d:\limit01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.LIMit = "test/limit01.csv"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.MMEMory.STORe.LIMit

Equivalent key

Analysis > **Limit Test** > **Edit Limit Line** > **Import from CSV File**

Equivalent SCPI command

Syntax

:MMEMory:LOAD:LIMit <string>

Example of use

```
10 OUTPUT 717;":MMEM:LOAD:LIM ""Test1/Limit01.csv"""
10 OUTPUT 717;":MMEM:LOAD:LIM ""D:Limit01.csv"""
```

SCPI.MMEMory.LOAD.RLIMit

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.LOAD.RLIMit = File

Description

This command recalls the specified ripple limit table file (file with the .csv extension saved with SCPI.MMEMory.STORe.RLIMit) of the active channel (specified with the SCPI.DISPlay.WINDow(Ch).ACTivate command), as ripple limit table for the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect).

NOTE Specify the file name with the extension. When you write directory names and file name, separate them with "/" (slash) or "\" (backslash).
If the specified file does not exist, an error occurs and the command is ignored.

Variable

Parameter	*File*
Description	File name of the ripple limit table (extension ".csv")
Data type	Character string type (String)
Range	254 characters or less
Note	If the specified file does not exist, an error occurs when executed and the object is ignored.

Examples

```
SCPI.DISPlay.WINDow(1).ACTive
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.RIMit = "D:\Rlimit01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTive
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.LOAD.RLIMit = "test/Rlimit01.csv"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.MMEMory.STORe.RLIMit

Equivalent key

Analysis > **Ripple Limit** > **Edit Ripple Line** > **Import from CSV File**

Equivalent SCPI command

Syntax

:MMEMory:LOAD:RLIMit <string>

Example of use

```
10 OUTPUT 717;":MMEM:LOAD:RLIM ""RTest1/Rlim01.csv"""
10 OUTPUT 717;":MMEM:LOAD:RLIM ""D:Rlim01.csv"""
```

SCPI.MMEMory.LOAD.SEGMent

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.LOAD.SEGMent = *File*

Description

This command recalls the specified segment sweep table file (file with a .csv extension saved with the SCPI.MMEMory.STORe.SEGMent), as the segment sweep table of the active channel.

NOTE Specify the file name with the extension. When you use directory names and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name of segment sweep table (extension ".csv")
Data type	Character string type (String)
Range	254 characters or less
Note	If the specified file does not exist, an error occurs when executed and the object is ignored.

Examples

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.LOAD.SEGMent = "d:\segm01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.LOAD.SEGMent = "test/segm01.csv"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.SEGMent

Equivalent key

Sweep Setup > **Edit Segment Table** > **Import from CSV File**

Equivalent SCPI command

Syntax

:MMEMory:LOAD:SEGMent <string>

Example of use

```
10 OUTPUT 717;":MMEM:LOAD:SEGM ""Test1/Segm01.csv"""
10 OUTPUT 717;":MMEM:LOAD:SEGM ""D:Segm01.csv"""
```

SCPI.MMEMory.LOAD.STATe

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.LOAD.STATe = *File*

Description

This command recalls the specified instrument state file (file with a .sta extension saved with SCPI.MMEMory.STORe.STATe).

NOTE Specify the file name with the extension. When you use directory names and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name of instrument state (extension ".sta")
Data type	Character string type (String)
Range	254 characters or less
Note	If the specified file does not exist, an error occurs when executed and the object is ignored.

Examples

SCPI.MMEMory.LOAD.STATe = "d:\state01.sta"

SCPI.MMEMory.LOAD.STATe = "test/state01.sta"

Related objects

SCPI.MMEMory.STORe.STATe

Equivalent key

Save/Recall > **Recall State** > **StateX** (X=01 to 08) | **AutoRec** | **File Dialog...** | **UserPres**

Equivalent SCPI command

Syntax

:MMEMory:LOAD[:STATe] <string>

Example of use

```
10 OUTPUT 717;":MMEM:LOAD ""Test1/State01.sta"""
10 OUTPUT 717;":MMEM:LOAD ""d:State01.sta"""
```

SCPI.MMEMory.MDIRectory

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.MDIRectory = *File*

Description

This command creates a new directory (folder).

NOTE When you create a directory under an existing directory, separate between the directory names with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	Directory name you want to create
Data type	Character string type (String)
Range	254 characters or less
Note	If a directory with the same name as the specified directory name exists, an error occurs when executed and the object is ignored.

Examples

```
SCPI.MMEMory.MDIRectory = "d:\test"
SCPI.MMEMory.MDIRectory = "test"
```

Equivalent key

Save/Recall > **Save State** > **File Dialog...**

Equivalent SCPI command

Syntax

```
:MMEMory:MDIRectory <string>
```

Example of use

```
10 OUTPUT 717;":MMEM:MDIR ""Test1"""""
10 OUTPUT 717;":MMEM:MDIR ""d:Test1"""""
```

SCPI.MMEMory.STORe.CHANnel.CLEar

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.CHANnel.CLEar

Description

This command deletes the instrument state for each channel (saved with the SCPI.MMEMory.STORe.CHANnel.STATe object) in all the registers.

Examples

SCPI.MMEMory.STORe.CHANnel.CLEar

Related objects

SCPI.MMEMory.STORe.CHANnel.STATe

Equivalent key

Save/Recall > **Save Channel** > **Clear States** > **OK**

Equivalent SCPI command

Syntax

:MMEMory:STORe:CHANnel:CLEar

Example of use

```
10 OUTPUT 717;":MMEM:STOR:CHAN:CLE"
```

SCPI.MMEMory.STORe.CHANnel.COEFficient

Object Type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.CHANnel.COEFficient = *Param*

Description

This command saves the channel coefficient data of the active channel specific to that channel only into the specified register (volatile memory).

Variable

Parameter	*Param*
Description	Register
Data Type	Character string type (String)
Range	Select from the following: • "A": Specifies register A. • "B": Specifies register B. • "C" :Specifies register C. • "D" :Specifies register D.
Note	If an instrument state has been already saved in the specified register, its contents are overwritten.

Examples

```
Dim Rgstr as String
Rgstr = "B"
SCPI.MMEMory.STORe.CHANnel.COEFficient = Rgstr
```

Related Objects

SCPI.MMEMory.LOAD.CHANnel.COEFficient

SCPI.MMEMory.LOAD.CHANnel.STATe

SCPI.MMEMory.STORe.CHANnel.STATe

Equivalent Key

Save/Recall > **Save Channel** > **Cal Only A** , **Cal Only B** , **Cal Only C** , **Cal Only D**

Equivalent SCPI Command

Syntax

:MMEMory:STORe:CHANnel:COEFficient {A|B|C|D}

Query Response

{A|B|C|D} <newline><^END>

Example of use

10 OUTPUT 717;":MMEM:STOR:CHAN:COEF C"

SCPI.MMEMory.STORe.CHANnel.STATe

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.CHANnel.STATe = *Register*

Description

This command saves the instrument state of the items set for the active channel specific to that channel only into the specified register (volatile memory).

Variable

Parameter	*Register*
Description	Register
Data type	Character string type (String)
Range	Select from the following: "A": Specifies register A. "B": Specifies register B. "C": Specifies register C. "D": Specifies register D.
Note	If an instrument state has been already saved in the specified register, its contents are overwritten.

Examples

SCPI.MMEMory.STORe.CHANnel.STATe = "a"

Related objects

SCPI.MMEMory.LOAD.CHANnel.STATe

SCPI.DISPlay.WINDow(Ch).ACTivate

Equivalent key

Save/Recall > **Save Channel** > **A|B|C|D**

Equivalent SCPI command

Syntax

:MMEMory:STORe:CHANnel[:STATe] {A|B|C|D}

Example of use

```
10 OUTPUT 717;":MMEM:STOR:CHAN A"
```

SCPI.MMEMory.STORe.EPARameters

Object Type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.EPARameters = *Value*

Description

This command saves the equivalent circuit parameters of an active channel into a CSV file at defined location.

Variable

Parameter	*Value*
Description	Filename and destination
Data Type	Character string type (String)
Range	254 chars
Preset Value	-
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var = "C:\sample.csv"
SCPI.MMEMory.STORe.EPARameters = Var
```

Equivalent Key

There is no equivalent key available on the front panel. However, similar key which exports to TXT file is shown below:

Analysis > **Equivalent Circuit** > **Export to TXT File...**

Equivalent SCPI Command

Syntax

:MMEMory:STORe:EPARameters

Example of use

```
10 OUTPUT 717;":MMEM:STOR:EPAR Sample_Text"
```

SCPI.MMEMory.STORe.FDATa

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.FDATa = *File*

Description

This command saves the formatted data array into a file in the CSV format (extension ".csv"), for the active trace of the active channel.

NOTE Specify the file name with the extension. When you use directory names and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name in which you want to save the formatted data array (extension ".csv")
Data type	Character string type (String)
Range	254 characters or less
Note	If a file with the same name as the specified file name exists, its contents are overwritten.

Examples

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.FDATa = "d:\trace01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.FDATa = "test/trace01.csv"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

Equivalent key

Save/Recall > **Save Trace Data**

Equivalent SCPI command

Syntax

:MMEMory:STORe:FDATa <string>

Example of use

```
10 OUTPUT 717;":MMEM:STOR:FDAT ""Result/Trace01.csv"""
10 OUTPUT 717;":MMEM:STOR:FDAT ""D:Trace01.csv"""
```

SCPI.MMEMory.STORe.IMAGe

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.IMAGe = *File*

Description

This command saves the display image on the LCD display at the execution of the object into a file in the bitmap (extension ".bmp") or portable network graphics (extension ".png") format.

NOTE When saving the E5061B measurement screen, execute the VBA program with the Visual Basic editor closed. For more information, see Running a Program from the E5061B Measurement Screen.

Specify the file name with the extension. When you use directory names and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name in which you want to save the display image on the LCD display (extension ".bmp" or ".png")
Data type	Character string type (String)
Range	254 characters or less
Note	If a file with the same name as the specified file name exists, its contents are overwritten.

Examples

```
SCPI.MMEMory.STORe.IMAGe = "d:\image01.bmp"
SCPI.MMEMory.STORe.IMAGe = "test/image01.png"
```

Equivalent key

System > **Dump Screen Image**

When performing the operation from the front panel, the image on the LCD display memorized in the volatile memory (clipboard) (the image on the LCD display when the **Capture** (**System**) key is pressed) is saved.

If no image is memorized in the clipboard, in the same way as the SCPI.MMEMory.STORe.IMAGe object, the image on the LCD display at the execution is memorized in the clipboard and then it is saved.

Equivalent SCPI command

Syntax

:MMEMory:STORe:IMAGe <string>

Example of use

```
10 OUTPUT 717;":MMEM:STOR:IMAG ""Result/Image01.bmp"""
10 OUTPUT 717;":MMEM:STOR:IMAG ""D:Image01.png"""
```

SCPI.MMEMory.STORe.LIMit

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.LIMit = *File*

Description

This command saves the limit table of the active trace of the active channel into a file in the CSV format (extension ".csv").

NOTE Specify the file name with the extension. When you use directory names and file name, separate them with "\ " (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name to save the limit table (extension ".csv")
Data type	Character string type (String)
Range	254 characters or less
Note	If a file with the same name as the specified file name exists, its contents are overwritten.

Examples

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.LIMit = "d:\limit01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.LIMit = "test/limit01.csv"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.MMEMory.LOAD.LIMit

Equivalent key

Analysis > **Limit Test** > **Edit Limit Line** > **Export to CSV File**

Equivalent SCPI command

Syntax

```
:MMEMory:STORe:LIMit <string>
```

Example of use

```
10 OUTPUT 717;":MMEM:STOR:LIM ""Test1/Limit01.csv"""
10 OUTPUT 717;":MMEM:STOR:LIM ""D:Limit01.csv"""
```

SCPI.MMEMory.STORe.RLIMit

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.RLIMit = File

Description

This command saves the ripple limit table of the active trace (specified with the SCPI.CALCulate(Ch).PARameter(Tr).SELect command) of the active channel (specified with the SCPI.DISPlay.WINDow(Ch).ACTivate command) into a file in the CSV format.

NOTE Specify the file name with the .sta extension. When you write directory names and file name, separate them with "/" (slash) or "\" (backslash).

If a file with the specified file name already exists, its contents are overwritten.

Variable

Parameter	*File*
Description	File name used to save the ripple limit table (extension ".csv")
Data type	Character string type (String)
Range	254 characters or less
Note	If the specified file does not exist, a runtime error occurs.

Examples (1)

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.RLIMit = "D:\Rlimit01.csv"
```

Examples (2)

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.CALCulate(1).PARameter(1).SELect
SCPI.MMEMory.STORe.RLIMit = "test/Rlimit01.csv"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.MMEMory.LOAD.RLIMit

Equivalent key

Analysis > **Ripple Limit** > **Edit Ripple Line** > **Export to CSV File**

Equivalent SCPI command

Syntax

:MMEMory:STORe:RLIMit <string>

Example of use

```
10 OUTPUT 717;":MMEM:STOR:RLIM ""RTest1/Rlim01.csv"""
10 OUTPUT 717;":MMEM:STOR:RLIM ""D:Rlim01.csv"""
```

SCPI.MMEMory.STORe.SALL

Object type

Property (**Read-Write**)

Syntax

SCPI.MMEMory.STORe.SALL = *Status*

Status = SCPI.MMEMory.STORe.SALL

Description

This command sets/gets whether to save the settings of all channels/traces or that of the displayed channels/traces only, as the instrument state to be saved.

Variable

Parameter	*Status*
Description	Selecting content to be saved as the instrument state setting.
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Specifies the setting of all channels/traces as the target to be saved. • False or OFF: Specifies the setting of displayed channels/traces only as the target to be saved.
Preset value	False or OFF

Examples

```
Dim Obj As Boolean
SCPI.MMEMory.STORe.SALL = True
Obj = SCPI.MMEMory.STORe.SALL
```

Related objects

SCPI.MMEMory.STORe.STATe

Equivalent key

Save/Recall > **Channel/Trace** > **Disp Only** | **ALL**

Equivalent SCPI command

Syntax

:MMEMory:STORe:SALL {ON|OFF|1|0}

:MMEMory:STORe:SALL?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":MMEM:STOR:SALL ON"
20 OUTPUT 717;":MMEM:STOR:SALL?"
30 ENTER 717;A
```

SCPI.MMEMory.STORe.SEGMent

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.SEGMent = *File*

Description

This command saves the segment sweep table of the active channel into a file in the CSV format (extension ".csv").

NOTE Specify the file name with the extension. When you use directory names and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name to save segment sweep table (extension ".csv")
Data type	Character string type (String)
Range	254 characters or less
Note	If a file with the same name as the specified file name exists, its contents are overwritten.

Examples

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.SEGMent = "d:\segm01.csv"
```

```
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.SEGMent = "test/segm01.csv"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.LOAD.SEGMent

Equivalent key

Sweep Setup > **Edit Segment Table** > **Export to CSV File**

Equivalent SCPI command

Syntax

:MMEMory:STORe:SEGMent <string>

Example of use

```
10 OUTPUT 717;":MMEM:STOR:SEGM ""Test1/Segm01.csv"""
10 OUTPUT 717;":MMEM:STOR:SEGM ""D:Segm01.csv"""
```

SCPI.MMEMory.STORe.SNP.DATA

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.SNP.DATA = *File*

Description

Saves the measurement data for the active channel (specified with the SCPI.DISPlay.WINDow(Ch).ACTivate command) into a file in the touchstone format.

You need to specify a file format and file type before saving a file. The extension differs depending on file types as shown in the following table:

<file type>	<extension>
When specifying one port	s1p
When specifying two ports	s2p

NOTE When you use directory names and file name, separate them with "/" (slash) or "\" (back slash).

If a file with the specified file name already exists, its contents are overwritten.

Variable

Parameter	*File*
Description	File name you want to use when saving file in the touchstone format
Range	254 characters or less

NOTE When invalid extension is specified, an error message appears and the command is ignored.

Examples

```
Dim SnpPorts(1) As Variant
SCPI.DISPlay.WINDow(1).ACTivate
SCPI.MMEMory.STORe.SNP.FORMat = "RI"
SnpPorts(0) = 1
```

```
SnpPorts(1) = 2
SCPI.MMEMory.STORe.SNP.TYPE.S2P = SnpPorts
SCPI.MMEMory.STORe.SNP.DATA = "SNP01.s2p"
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.SNP.FORMat

SCPI.MMEMory.STORe.SNP.TYPE.S1P

SCPI.MMEMory.STORe.SNP.TYPE.S2P

Equivalent key

After a file type is specified, a dialog box will appear.

Equivalent SCPI command

Syntax

:MMEMory:STORe:SNP[:DATA] <string>

Example of use

```
10 OUTPUT 717;"DISP:WIND1:ACT"
20 OUTPUT 717;":MMEM:STOR:SNP:FORM RI"
30 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S2P 1,2"
40 OUTPUT 717;":MMEM:STOR:SNP ""SNP01.s2p"""
```

SCPI.MMEMory.STORe.SNP.FORMat

Object type

Property (**Read-Write**)

Syntax

SCPI.MMEMory.STORe.SNP.FORMat = *Param*

Param = SCPI.MMEMory.STORe.SNP.FORMat

Description

This command sets/gets the data format for saving measurement data for the active channel (specified with SCPI.DISPlay.WINDow(Ch).ACTivate command) into a file in the touchstone format.

Variable

Parameter	*Param*
Description	Touchstone file format
Data type	Character string type (String)
Range	Select from the following: "AUTO": Specifies data format automatically according to the display format of the active trace. "MA": Specifies data format "log magnitude > angle". "DB": Specifies data format "linear magnitude > angle". "RI": Specifies data format "real part > imaginary part".
Preset value	"AUTO"

Examples

```
Dim Fmt As String
SCPI.MMEMory.STORe.SNP.FORMat = "MA"
Fmt = SCPI.MMEMory.STORe.SNP.FORMat
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.SNP.DATA

Equivalent key

Save/Recall > **Save Snp** > **Snp Format** > **AUTO|LogMag/Angle|LinMag/Angle|Real/Imaginary**

Equivalent SCPI command

Syntax

:MMEMory:STORe:SNP:FORMat {AUTO|MA|DB|RI}

:MMEMory:STORe:SNP:FORMat?

Query response

{AUTO|MA|DB|RI}<newline><^END>

Example of use

```
10 OUTPUT 717;":MMEM:STOR:SNP:FORM MA"
20 OUTPUT 717;":MMEM:STOR:SNP:FORM?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.SNP.TYPE.S1P

Object type

Property (**Read-Write**)

Syntax

SCPI.MMEMory.STORe.SNP.TYPE.S1P = *Port*

Port = SCPI.MMEMory.STORe.SNP.TYPE.S1P

Description

This command sets/gets the specified port to the file type (1 port) when saving measurement data for the active channel (specified with SCPI.DISPlay.WINDow(Ch).ACTivate command) into a file in the touchstone format.

Variable

Parameter	*Port*
Description	Port number
Range	1 to 2
Resolution	1

Examples

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P 2"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P?"
30 ENTER 717;A$
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.SNP.DATA

SCPI.MMEMory.STORe.SNP.FORMat

Equivalent key

Save/Recall > **Save SnP** > **S1P** > **1|2**

Equivalent SCPI command

Syntax

:MMEMory:STORe:SNP:TYPE:S1P <numeric>

:MMEMory:STORe:SNP:TYPE:S1P?

Example of use

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P 2"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S1P?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.SNP.TYPE.S2P

Object type

Property (**Read-Write**)

Syntax

SCPI.MMEMory.STORe.SNP.TYPE.S2P = *Ports*

Ports = SCPI.MMEMory.STORe.SNP.TYPE.S2P

Description

This command sets/gets the specified port to the file type (2 port) when saving measurement data for the active channel (specified with SCPI.DISPlay.WINDow(Ch).ACTivate command) into a file in the touchstone format.

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies a port for file type. • *Ports(1):* Specifies the other port for file type. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	If the specified variable is out of the allowable setup range, an error occurs when it is executed. If you specify the same port number to two ports, an error occurs during execution. The order of the two port numbers to be specified is arbitrary.

Examples

```
Dim Ports(1) As Long
Ports(0) = 1
Ports(1) = 2
SCPI.MMEMory.STORe.SNP.TYPE.S2P = Ports
Ports = SCPI.MMEMory.STORe.SNP.TYPE.S2P
```

Related objects

SCPI.DISPlay.WINDow(Ch).ACTivate

SCPI.MMEMory.STORe.SNP.DATA

SCPI.MMEMory.STORe.SNP.FORMat

Equivalent key

Save/Recall > **Save Snp** > **S2p** > **1-2**

Equivalent SCPI command

Syntax

:MMEMory:STORe:SNP:TYPE:S2P <numeric1>, <numeric 2>

:MMEMory:STORe:SNP:TYPE:S2P?

Example of use

```
10 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S2P 1,2"
20 OUTPUT 717;":MMEM:STOR:SNP:TYPE:S2P?"
30 ENTER 717;A$
```

SCPI.MMEMory.STORe.STATe

Object type

Method (**Write Only**)

Syntax

SCPI.MMEMory.STORe.STATe = *File*

Description

This command saves the instrument state (contents to be saved specified with the SCPI.MMEMory.STORe.STYPe object) into a file (file with the .sta extension).

NOTE Specify the file name with the extension. When you use directory names and file name, separate them with "\" (back slash), or "/" (slash).

Variable

Parameter	*File*
Description	File name to save the instrument state (extension ".sta")
Data type	Character string type (String)
Range	254 characters or less
Note	If a file with the same name as the specified file name exists, its contents are overwritten.

Examples

```
Dim StaType As String
SCPI.MMEMory.STORe.STYPe = "cdst"
SCPI.MMEMory.STORe.STATe = "d:\state01.sta"
```

```
Dim StaType As String
SCPI.MMEMory.STORe.STYPe = "cdst"
SCPI.MMEMory.STORe.STATe = "test/state01.sta"
```

Related objects

SCPI.MMEMory.STORe.STYPe

SCPI.MMEMory.LOAD.STATe

Equivalent key

Save/Recall > **Save State** > **StateX** (X=01 to 08) | **AutoRec** | **File Dialog...** | **UserPres**

Equivalent SCPI command

Syntax

```
:MMEMory:STORe[:STATe] <string>
```

Example of use

```
10 OUTPUT 717;":MMEM:STOR ""Test1/State01.sta"""
10 OUTPUT 717;":MMEM:STOR ""D:State01.sta"""
```

SCPI.MMEMory.STORe.STYPe

Object type

Property (**Read-Write**)

Syntax

SCPI.MMEMory.STORe.STYPe = *Param*

Param = SCPI.MMEMory.STORe.STYPe

Description

Selects the contents saved when saving the instrument state into a file with the SCPI.MMEMory.STORe.STATe object.

Variable

Parameter	*Param*
Description	Data of instrument state
Data type	Character string type (String)
Range	Select from the following. • "STATe": Specifies the save of the measurement conditions only. • "CSTate":Specifies the save of the measurement conditions and the calibration state. • "DSTate": Specifies the save of the measurement conditions and the formatted data array. • "CDSTate": Specifies the save of the measurement conditions, the calibration state, and the formatted data array.
Preset value	"CSTate"

Examples

```
Dim StaType As String
SCPI.MMEMory.STORe.STYPe = "cdst"
StaType = SCPI.MMEMory.STORe.STYPe
```

Related objects

SCPI.MMEMory.STORe.STATe

Equivalent key

Save/Recall > Save Type > State Only|State & Cal|State & Trace|All

Equivalent SCPI command

Syntax

```
:MMEMory:STORe:STYPe {STATe|CSTate|DSTate|CDSTate}
:MMEMory:STORe:STYPe?
```

Query response

```
{STAT|CST|DST|CDST}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":MMEM:STOR:STYP CDST"
20 OUTPUT 717;":MMEM:STOR:STYP?"
30 ENTER 717;A$
```

OUTPUT

SCPI.OUTPut.COUPling

Object Type

Property (**Read-Write**)

Syntax

SCPI.OUTPut.COUPling = *Param*

Param = SCPI.OUTPut.COUPling

Description

This command sets/gets the port coupling type whether the signal is AC or DC to the output port.

Variable

Parameter	*Param*
Description	Port coupling type
Data Type	Character string type (String)
Range	Select from the following: • "AC": Specifies the AC coupling. • "DC": Specifies the DC coupling.
Preset Value	"DC"

Examples

```
Dim PtType as String
SCPI.OUTPut.COUPling = "DC"
PtType = SCPI.OUTPut.COUPling
```

Related Objects

Equivalent Key

System > **S-Param Port Couple** > **AC|DC**

Equivalent SCPI Command

Syntax

:OUTPut:COUPling {AC|DC}

:OUTPut:COUPling?

Query Response

{AC|DC} <newline><^END>

Example of use

```
10 OUTPUT 717;":OUTP:COUP DC"
20 OUTPUT 717;":OUTP:COUP?"
30 ENTER 717;A$
```

SCPI.OUTPut.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.OUTPut.STATe = *Status*

Status = SCPI.OUTPut.STATe

Description

This command turns ON/OFF of the stimulus signal output. Measurement cannot be made until the stimulus signal output is turned ON.

Variable

Parameter	*Status*
Description	ON/OFF of the stimulus signal output
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns on the stimulus signal. • False or OFF: Turns off the stimulus signal.
Preset value	True or ON

Examples

```
Dim Outp As Boolean
SCPI.OUTPut.STATe = True
Outp = SCPI.OUTPut.STATe
```

Equivalent key

Sweep Setup > **Power** > **RF Out**

System > **Overload Recovery** > **RF Out**

Equivalent SCPI command

Syntax

:OUTPut[:STATe] {ON|OFF|1|0}

:OUTPut[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":OUTP ON"
20 OUTPUT 717;":OUTP?"
30 ENTER 717;A
```

PROGRAM

:PROGram:CATalog?

No equivalent COM Commands

Syntax

:PROGram:CATalog?

Description

This command reads the list of all executable VBA macros (procedures defined by Public including the VBA project loaded on the VBA editor).

Query response

{string}<newline><^END>

The character string in the following format, in which each macro is separated by a comma (.), is read out.

"{macro 1},{macro 2}, ... ,{macro N}"

Where N is the total number of VBA macros.

{macro n}: VBA macro name (module name.procedure name)

Example of use

```
10 DIM A$[1000]
20 OUTPUT 717;":PROG:CAT?"
30 ENTER 717;A$
```

:PROGram[:SELected]:NAME

No equivalent COM Commands

Syntax

:PROGram[:SELected]:NAME <string>

:PROGram[:SELected]:NAME?

Description

This command sets/gets the VBA macro controlled with the :PROG:STAT command.

NOTE Selectable VBA macro names can be read with the :PROG:CAT? command.

Variable

Parameter	*String*
Description	VBA macro name (module name.procedure name)
Range	254 characters or less
Preset value	""

Query response

{string}<newline><^END>

Example of use

```
10 OUTPUT 717;":PROG:NAME ""Module1.main"""
20 OUTPUT 717;":PROG:NAME?"
30 ENTER 717;A$
```

Related commands

:PROG:CAT?

:PROG:STAT

Equivalent key

Macro Setup > **Select Macro**

NOTE When performing a similar operation from the front panel, you should select the VBA macro and execute it at the same.

:PROGram[:SELected]:STATe

No equivalent COM Commands

Syntax

:PROGram[:SELected]:STATe {STOP|RUN}

:PROGram[:SELected]:STATe?

Description

This command sets/gets the control/state of the VBA macro selected with the :PROG:STAT command.

Variable

Range	STOP: Stop the program. RUN: Run the Program.
Preset value	STOP

Query response

{STOP|RUN}<newline><^END>

Example of use

```
10 OUTPUT 717;":PROG:STAT RUN"
20 OUTPUT 717;":PROG:STAT?"
30 ENTER 717;A$
```

Related commands

:PROG:NAME

Equivalent key

Macro Break (to stop)

Macro Setup > **Select Macro** (to run)

NOTE When performing the operation from the front panel, you select the VBA macro and execute it at the same time.

SCPI.PROGram.VARiable.ARRay(*Vnum*).DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.PROGram.VARiable.ARRay(*Vnum*).DATA = *Data*

Data = SCPI.PROGram.VARiable.ARRay(*Vnum*).DATA

Description

This command sets/gets an array of Data that can be exchanged between an external PC and E5061B built-in VBA using GPIB/LAN/USB.

Variable

Parameter	*Vnum*
Description	Variable Number
Range	1 to 10
Preset value	1

Parameter	*Data*
Description	"n" is the number obtained from the value specified with the SCPI.PROGram.VARiable.ARRay(Vnum).SIZE object. • Data(0): The first array data • Data(n-1): The n-th array data The index of the array starts from 0.
Data Type	Variant type (Variant)
Note	If the data size is not specified, an error occurs when executed.

Examples

```
Dim Var1(2) As Long
Dim Var2 as Variant
Dim ArraySize as Long
ArraySize=3
```

```
Var1(1) = 2
Var1(2) = 56
SCPI.PROGram.VARiable.ARRay(1).SIZE = ArraySize
SCPI.PROGram.VARiable.ARRay(1).DATA = Var1
Var2= SCPI.PROGram.VARiable.ARRay(1).DATA
```

Related objects

SCPI.PROGram.VARiable.ARRay(Vnum).SIZE

SCPI.PROGram.VARiable.DOUBle(Vnum).DATA

SCPI.PROGram.VARiable.LONG(Vnum).DATA

SCPI.PROGram.VARiable.STRing(Vnum).DATA

Equivalent key

None

Equivalent SCPI command

Syntax

:PROGram:VARiable:ARRay{[1] - 10}[:DATA] <Data Array>

:PROGram:VARiable:ARRay{[1] - 10}[:DATA]?

Example of use

```
10 OUTPUT 717;"PROG:VAR:ARR2:SIZE 4"
20 OUTPUT 717;"PROG:VAR:ARR2 1.0,2.0,3.0,4.0"
30 OUTPUT 717;"PROG:VAR:ARR2?"
40 ENTER 717;A(*)
```

SCPI.PROGram.VARiable.ARRay(*Vnum*).SIZE

Object type

Property (**Read-Write**)

Syntax

SCPI.PROGram.VARiable.ARRay(*Vnum*).SIZE = *Value*

Value = SCPI.PROGram.VARiable.ARRay(*Vnum*).SIZE

Description

This command sets/gets the size of an array of Data, specified by SCPI.PROGram.VARiable.ARRay(1-10).DATA, that can be exchanged between an external PC and E5061B built-in VBA using GPIB/LAN/USB.

Variable

Parameter	*Vnum*
Description	Variable Number
Range	1 to 10
Preset value	1

Parameter	*Value*
Description	The value of data size
Data Type	Long integer type (long)
Range	1 to 40002
Preset value	402
Resolution	1

Examples

```
Dim Var1(2) As Long
Dim Var2 as Variant

Dim ArraySize as Long
```

ArraySize=3

Var1(1) = 2
Var1(2) = 56
SCPI.PROGram.VARiable.ARRay(1).SIZE = ArraySize
SCPI.PROGram.VARiable.ARRay(1).DATA = Var1
Var2= SCPI.PROGram.VARiable.ARRay(1).DATA

Related objects

SCPI.PROGram.VARiable.ARRay(Vnum).DATA

SCPI.PROGram.VARiable.DOUBle(Vnum).DATA

SCPI.PROGram.VARiable.LONG(Vnum).DATA

SCPI.PROGram.VARiable.STRing(Vnum).DATA

Equivalent key

None

Equivalent SCPI command

Syntax

:PROGram:VARiable:ARRay{[1] - 10}:SIZE <Data Array Size>

:PROGram:VARiable:ARRay{[1] - 10}:SIZE ?

Example of use

```
10 OUTPUT 717;"PROG:VAR:ARR2:SIZE 4"
20 OUTPUT 717;"PROG:VAR:ARR2 1.0,2.0,3.0,4.0"
30 OUTPUT 717;"PROG:VAR:ARR2?"
40 ENTER 717;A(*)
```

SCPI.PROGram.VARiable.DOUBle(*Vnum*).DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.PROGram.VARiable.DOUBle(*Vnum*).DATA = *Value*

Value = SCPI.PROGram.VARiable.DOUBle(*Vnum*).DATA

Description

This command can be used to exchange Double type data between an external PC and E5061B built-in VBA using GPIB/LAN/USB.

Variable

Parameter	*Vnum*
Description	Variable Number
Range	1 to 10
Preset value	1

Parameter	*Value*
Description	The value of the double precision floating point type
Data type	Double precision floating point type (Double)
Range	Compliant with the double precision floating point type
Preset value	0

Examples

```
Dim DblVal As Double
DblVal =55.7890

SCPI.PROGram.VARiable.DOUBle(1).DATA = DblVal
DblVal= SCPI.PROGram.VARiable.DOUBle(1).DATA
```

Related objects

SCPI.PROGram.VARiable.ARRay(Vnum).DATA

SCPI.PROGram.VARiable.ARRay(Vnum).SIZE

SCPI.PROGram.VARiable.LONG(Vnum).DATA

SCPI.PROGram.VARiable.STRing(Vnum).DATA

Equivalent key

None

Equivalent SCPI command

Syntax

:PROGram:VARiable:DOUBle{[1]-10}[:DATA] <Numeric value, Double Type>

:PROGram:VARiable:DOUBle{[1]-10}[:DATA]?

Example of use

```
10 OUTPUT 717;"PROGram:VARiable:DOUBle1 12345.89607"
20 OUTPUT 717;":PROGram:VARiable:DOUBle1?"
30 ENTER 717;A$
```

SCPI.PROGram.VARiable.LONG(*Vnum*).DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.PROGram.VARiable.LONG(*Vnum*).DATA = *Value*

Value = SCPI.PROGram.VARiable.LONG(*Vnum*).DATA

Description

This command can be used to exchange Long type data between an external PC and E5061B built-in VBA using GPIB/LAN/USB.

Variable

Parameter	*Vnum*
Description	Variable Number
Range	1 to 10
Preset value	1

Parameter	*Value*
Description	The value of the long integer type
Data type	Long integer type (Long)
Range	Compliant with the long integer type
Preset value	0

Examples

```
Dim LngVal As Long
LngVal =5512345
SCPI.PROGram.VARiable.LONG(1).DATA = LngVal
LngVal= SCPI.PROGram.VARiable.LONG(1).DATA
```

Related objects

SCPI.PROGram.VARiable.ARRay(Vnum).DATA

SCPI.PROGram.VARiable.ARRay(Vnum).SIZE

SCPI.PROGram.VARiable.DOUBle(Vnum).DATA

SCPI.PROGram.VARiable.STRing(Vnum).DATA

Equivalent key

None

Equivalent SCPI command

Syntax

:PROGram:VARiable:LONG{[1] - 10}:DATA <Numeric value, Long Type>

:PROGram:VARiable:LONG{[1] - 10}:DATA?

Example of use

```
10 OUTPUT 717;":PROGram:VARiable:LONG1:DATA 123459607"
20 OUTPUT 717;":PROGram:VARiable:LONG1:DATA?"
30 ENTER 717;A$
```

SCPI.PROGram.VARiable.STRing(*Vnum*).DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.PROGram.VARiable.STRing(*Vnum*).DATA = *Value*

Value = SCPI.PROGram.VARiable.STRing(*Vnum*).DATA

Description

This command can be used to exchange String type data between an external PC and E5061B built-in VBA using GPIB/LAN/USB.

Variable

Parameter	*Vnum*
Description	Variable Number
Range	1 to 10
Preset value	1

Parameter	*Value*
Description	The value of the character string type
Data type	Character string type (String)
Range	Compliant with the character string type
Preset value	""

Examples

```
Dim StrVal As String
StrVal = "TestS11"

SCPI.PROGram.VARiable.STRing(1).DATA = StrVal
StrVal= SCPI.PROGram.VARiable.String(1).DATA
```

Related objects

SCPI.PROGram.VARiable.ARRay(Vnum).DATA

SCPI.PROGram.VARiable.ARRay(Vnum).SIZE

SCPI.PROGram.VARiable.DOUBle(Vnum).DATA

SCPI.PROGram.VARiable.LONG(Vnum).DATA

Equivalent key

None

Equivalent SCPI command

Syntax

:PROGram:VARiable:STRing{[1] - 10}[:DATA] <Character value, String Type>

:PROGram:VARiable:STRing{[1] - 10}[:DATA]?

Example of use

```
10 OUTPUT 717;" PROG:VAR:STR ""TEST DATA"""
20 OUTPUT 717;" PROG:VAR:STR?"
30 ENTER 717;A$
```

SENSE

SCPI.SENSe(*Ch*).AVERage.CLEar

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).AVERage.CLEar

Description

This command resets the data count to 0, it is used for averaging the selected channel (*Ch*). Measurement data before the execution of this object is not used for averaging.

Examples

SCPI.SENSe(1).AVERage.CLEar

Related objects

SCPI.SENSe(Ch).AVERage.COUNt

SCPI.SENSe(Ch).AVERage.STATe

Equivalent key

Avg > Averaging Restart

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:AVERage:CLEar

Example of use

```
10 OUTPUT 717;":SENS1:AVER:CLE"
```

SCPI.SENSe(*Ch*).AVERage.COUNt

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).AVERage.COUNt = *Value*

Value = SCPI.SENSe(*Ch*).AVERage.COUNt

Description

This command sets/gets the averaging factor of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Averaging factor
Data type	Long integer type (Long)
Range	1 to 999
Preset value	16
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim AvgCnt As Long
SCPI.SENSe(1).AVERage.COUNt = 4
AvgCnt = SCPI.SENSe(1).AVERage.COUNt
```

Related objects

SCPI.SENSe(Ch).AVERage.STATe

SCPI.SENSe(Ch).AVERage.CLEar

Equivalent key

Avg > **Avg Factor**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:AVERage:COUNt <numeric>

:SENSe{[1]-4}:AVERage:COUNt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:AVER:COUN 4"
20 OUTPUT 717;":SENS1:AVER:COUN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).AVERage.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).AVERage.STATe = *Status*

Status = SCPI.SENSe(*Ch*).AVERage.STATe

Description

This command sets/gets the averaging function of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the averaging function
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON(1): Turns ON • False or OFF(0): Turns OFF
Preset value	False or OFF

Examples

```
Dim Avg As Boolean
SCPI.SENSe(1).AVERage.STATe = True
Avg = SCPI.SENSe(1).AVERage.STATe
```

Related objects

SCPI.SENSe(Ch).AVERage.COUNt

SCPI.SENSe(Ch).AVERage.CLEar

Equivalent key

Avg > **Averaging**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:AVERage[:STATe] {ON|OFF|1|0}

:SENSe{[1]-4}:AVERage[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:AVER ON"
20 OUTPUT 717;":SENS1:AVER?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).BANDwidth.RESolution

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).BANDwidth.RESolution = *Value*

Value = SCPI.SENSe(*Ch*).BANDwidth.RESolution

Description

This command sets/gets the IF bandwidth of the selected channel (*Ch*).

NOTE This command is similar to SCPI.SENSe(Ch).BWIDth.RESolution.

Variable

Parameter	*Value*
Description	IF bandwidth
Data type	Double precision floating point type (Double)
Range	1 to 300000 (1|1.5|2|3|4|5|7|10|15|20|30|40|50|70|100| 150|200|300|400|500|700|1k|1.5k|2k|3k|4k| 5k|7k|10k|15k|20k|30k|40k|50k|70k|100k| 150k|200k|300k)
Preset value	30000
Unit	Hz (hertz)
Resolution	Refer the Range.
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim IfBw As Double
SCPI.SENSe(1).BANDwidth.RESolution = 1.5E3
IfBw = SCPI.SENSe(1).BANDwidth.RESolution
```

Related objects

SCPI.SENSe(Ch).BWIDth.RESolution

Equivalent key

Avg > **IF Bandwidth**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:BANDwidth[:RESolution] <numeric>
:SENSe{[1]-4}:BANDwidth[:RESolution]?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:BAND 1.5E3"
20 OUTPUT 717;":SENS1:BAND?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).BWAuto.LIMit.RESolution

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).BWAuto.LIMit.RESolution = *Value*

Value = SCPI.SENSe(*Ch*).BWAuto.LIMit.RESolution

Description

This command sets/gets maximum IF bandwidth in bandwidth auto mode, for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	maximum IF bandwidth in bandwidth auto mode
Data Type	Double precision floating point type (Double)
Range	1 to 3E+5 (1|1.5|2|3|4|5|7|10|15|20|30|40|50|70|100| 150|200|300|400|500|700|1k|1.5k|2k|3k|4k| 5k|7k|10k|15k|20k|30k|40k|50k|70k|100k| 150k|200k|300k)
Preset Value	3E+4
Unit	Hz

Examples

```
Dim BWALim as Double
SCPI.SENSe(4).BWAuto.LIMit.RESolution = 3E+4
BWALim = SCPI.SENSe(4).BWAuto.LIMit.RESolution
```

Related Objects

SCPI.SENSe(Ch).BWAuto.STATus

Equivalent Key

Avg > **IFBW Auto Limit**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:BWAuto:LIMit[:RESolution] <numeric>

:SENSe{[1]-4}:BWAuto:LIMit[:RESolution]?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:BWA:LIM 1E+5"
20 OUTPUT 717;":SENS1:BWA:LIM?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).BWAuto.STATus

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).BWAuto.STATus = *Status*

Status = SCPI.SENSe(*Ch*).BWAuto.STATus

Description

This command turns ON/OFF the IF bandwidth auto function, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the IF bandwidth auto function
Data Type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON(1): Turns ON • False or OFF(0): Turns OFF
Preset Value	False or OFF

Examples

```
Dim StatBWA as Boolean
SCPI.SENSe(4).BWAuto.STATus = True
StatBWA = SCPI.SENSe(4).BWAuto.STATus
```

Related Objects

SCPI.SENSe(Ch).BWAuto.LIMit.RESolution

Equivalent Key

Avg > **IFBW Auto**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:BWAuto[:STATus] {ON|OFF|1|0}

:SENSe{[1]-4}:BWAuto[:STATus]?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:BWA OFF"
20 OUTPUT 717;":SENS1:BWA?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).BWIDth.RESolution

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).BWIDth.RESolution = *Value*

Value = SCPI.SENSe(*Ch*).BWIDth.RESolution

Description

This command sets/gets the IF bandwidth of the selected channel (*Ch*).

NOTE This command is similar to SCPI.SENSe(Ch).BANDwidth.RESolution.

Variable

Parameter	*Value*
Description	IF bandwidth
Data type	Double precision floating point type (Double)
Range	1 to 300000 (1|1.5|2|3|4|5|7|10|15|20|30|40|50|70|100| 150|200|300|400|500|700|1k|1.5k|2k|3k|4k| 5k|7k|10k|15k|20k|30k|40k|50k|70k|100k| 150k|200k|300k)
Preset value	30000
Unit	Hz (hertz)
Resolution	Refer the Range.
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim IfBw As Double
SCPI.SENSe(1).BWIDth.RESolution = 1.5E3
IfBw = SCPI.SENSe(1).BWIDth.RESolution
```

Related objects

SCPI.SENSe(Ch).BANDwidth.RESolution

Equivalent key

Avg > **IF Bandwidth**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:BWIDth[:RESolution] <numeric>

:SENSe{[1]-4}:BWIDth[:RESolution]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:BWID 1.5E3"
20 OUTPUT 717;":SENS1:BWID?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.CLEar

Type of object

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.CLEar

Description

This command clears the error coefficient for calibration when the frequency offset feature is set to OFF, for the selected channel (*Ch*).

This command also clears the fixture compensation data when option 005 is installed.

Example of use

SCPI.SENSe(1).CORRection.CLEar

Related objects

SCPI.SENSe(Ch).OFFSet.STATe

SCPI.SENSe(Ch).CORRection.OFFSet.CLEar

Equivalent key

Cal > **Clear** > **OK**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:CLEar

Example of use

```
10 OUTPUT 717;":SENS1:CORR:CLE"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.DATA (*Str*, *Int1*, *Int2*) = *Array*

Array = SCPI.SENSe(*Ch*).CORRection.COEFficient.DATA (*Str*, *Int1*, *Int2*)

Description

This command sets/gets the calibration coefficient data for the specified channel.

NOTE When the calibration factor is interpolated, the interpolated calibration coefficient array is read. Similarly, when the calibration factor is not interpolated, a non-interpolated calibration coefficient array is read.

After writing the calibration coefficient array, the written value becomes effective only after the (SCPI.SENSe(Ch).CORRection.COEFficient.SAVE) command is executed.

Variable

Parameter	*Array*
Description	Indicates the array data (corrected data array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. *Data(n×2-2)* Real part of data (complex number) at the n-th measurement point. *Data(n×2-1)* Imaginary part of data (complex number) at the n-th measurement point. The index of the array starts from 0.
Data type	Variant type (Variant)

Parameter	*Str*
Description	Calibration type
Data type	Character string type (String)
Range	Select from the following: • "ES": Source match • "ER": Reflection tracking • "ED": Directivity • "EL": Load match • "ET": Transmission tracking • "EX": Isolation

Parameter	*Int1*
Description	Response port
Data type	Integer type (Integer)
Range	1 or 2
Note	If ES, ER, or ED is used, the response port and the stimulus port must be the same, while EL, ET, or EX is used, the response port and the stimulus port must be different.

Parameter	*Int2*

Description	Stimulus port
Data type	Integer type (Integer)
Range	1 or 2
Note	If ES, ER, or ED is used, the response port and the stimulus port must be the same, while EL, ET, or EX is used, the response port and the stimulus port must be different.

For information on the variable (*Ch*), see Ch

Examples

```
DIM CoefArray As Variant
CoefArray = SCPI.SENSe(1).CORRection.COEFficient.DATA("EL", 1, 2)


SCPI.SENSe(2).CORRection.COEFficient.DATA("EL", 1, 2) = CoefArray
SCPI.SENSe(2).CORRection.COEFficient.SAVE
```

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.ERESponse

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.OPEN

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.SHORt

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.THRU

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT1

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT2

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient[:DATA]
{ES|ER|ED|EL|ET|EX},<numeric 1>, <numeric 2>, <numeric 3>,...,
<numeric 3 n×2>

:SENSe{[1]-4}:CORRection:COEFficient[:DATA]?
{ES|ER|ED|EL|ET|EX},<numeric 1>, <numeric 2>

Query response

{numeric 1}, ... ,{numeric NOP×2}<newline><^END>

	Description
{numeric n×2-1}	Real part of data (complex number) at the n-th measurement point.
{numeric n×2}	Imaginary part of data (complex number) at the n-th measurement point.

Because the calibration coefficient array is expressed by a complex number, the real part and the imaginary part of one measurement point are returned and obtained as a value. Here, NOP is the number of measurement points and n is an integer between 1 and NOP.

Example of use

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:CORR:COEF? EL,1,2"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.GPData

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.GPData(*Str*) = *Array*

Array = SCPI.SENSe(*Ch*).CORRection.COEFficient.GPData(*Str*)

Description

This command sets/gets the calibration coefficient data of Gain-Phase measurement, for the selected channel (*Ch*).

NOTE When the calibration factor is interpolated, the interpolated calibration coefficient array is read. Similarly, when the calibration factor is not interpolated, a non-interpolated calibration coefficient array is read.

After writing the calibration coefficient array, the written value becomes effective only after the (SCPI.SENSe(Ch).CORRection.COEFficient.SAVE) command is executed.

Variable

Parameter	*Array*
Description	Indicates the array data (corrected data array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. *Data(n×2-2)* Real part of data (complex number) at the n-th measurement point. *Data(n×2-1)* Imaginary part of data (complex number) at the n-th measurement point. The index of the array starts from 0.
Data type	Variant type (Variant)

Parameter	*Str*
Description	Calibration coefficient type
Data type	Character string type (String)
Range	Select from the following: • "ES": Source match • "ER": Reflection tracking • "ED": Directivity • "EL": Load match • "ET": Transmission tracking • "EX": Isolation

Examples

```
Dim GpCorr as Variant
GpCorr = SCPI.SENSe(1).CORRection.COEFficient.GPData("ES")

SCPI.SENSe(2).CORRection.COEFficient.GPData("ES") = GpCorr
SCPI.SENSe(2).CORRection.COEFficient.SAVE
```

Related Objects

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.OPEN

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.SHORt

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.THRU

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPS1

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:GPData {ES|ER|ED|EL|ET|EX}

:SENSe{[1]-4}:CORRection:COEFficient:GPData?

Query Response

{numeric 1}, ... ,{numeric NOP×2}<newline><^END>

	Description
{numeric n×2-1}	Real part of data (complex number) at the n-th measurement point.
{numeric n×2}	Imaginary part of data (complex number) at the n-th measurement point.

Example of use

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:CORR:COEF:GPD? EL"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.ERESponse

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.ERESponse = *Ports*

Description

This command sets the calibration type to the enhanced response calibration between the two specified ports when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.DATA command, for the selected channel (*Ch*).

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies the response port. • *Ports(1):* Specifies the stimulus port. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	For each parameter, you must specify a different port number. If you specify the same port number for 2 or more parameters, an error occurs and the command is ignored.

Examples

```
Dim ERESPort(1) As Variant
ERESPort(0) = 1
ERESPort(1) = 2
SCPI.SENSe(1).CORRection.COEFficient.METHod.ERESponse = ERESPort
```

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod:ERESponse <numeric 1>,<numeric 2>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:ERES 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPResponse.OPEN

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPResponse.OPEN

Description

This command sets the calibration type to the response calibration (open) of the gain phase port when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.GPData command, for the selected channel (*Ch*).

Examples

SCPI.SENSe(2).CORRection.COEFficient.METHod.GPResponse.OPEN

Related Objects

SCPI.SENSe(Ch).CORRection.COEFficient.GPData

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.SHORt

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.THRU

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPS1

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod:GPResponse:OPEN

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:GPR:OPEN"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPResponse.SHORt

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPResponse.SHORt

Description

This command sets the calibration type to the response calibration (short) of the gain phase port when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.GPData command, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COEFficient.METHod.GPResponse.SHORt

Related Objects

SCPI.SENSe(Ch).CORRection.COEFficient.GPData

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.OPEN

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.THRU

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPS1

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod:GPResponse:SHORt

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:GPR:SHOR"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPResponse.THRU

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPResponse.THRU

Description

This command sets the calibration type to the response calibration (thru) between the T-port and the R-port using gain phase measurement when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.GPData command, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COEFficient.METHod.GPResponse.THRU

Related Objects

SCPI.SENSe(Ch).CORRection.COEFficient.GPData

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.OPEN

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.SHORt

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPS1

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod:GPResponse:THRU

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:GPR:THRU"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPS1

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.GPS1

Description

This command sets the calibration type to the full 1-port SOLT calibration of the T-port using gain phase measurement, when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.GPData command, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COEFficient.METHod.GPS1

Related Objects

SCPI.SENSe(Ch).CORRection.COEFficient.GPData

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.OPEN

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.SHORt

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.GPResponse.THRU

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod:GPS1

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:GPS1"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.RESPonse.OPEN

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.RESPonse.OPEN = *Port*

Description

This command sets the calibration type to the response calibration (open) of the specified port when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.DATA command, for the selected channel (*Ch*).

Variable

See *Port*.

Examples

SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.OPEN = 1

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod[:RESPonse]:OPEN <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:OPEN 1"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.RESPonse.SHORt

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.RESPonse.SHORt = *Port*

Description

This command sets the calibration type to the response calibration (short) of the specified port for the selected channel, when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.DATA command.

Variable

See *Port*.

Examples

SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.SHORt = 1

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod[:RESPonse]:SHORt <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:SHOR 1"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.RESPonse.THRU

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.RESPonse.THRU = *Ports*

Description

This command sets the calibration type to the response calibration (thru) between the two specified ports when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.DATA command, for the selected channel (*Ch*)

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies the response port. • *Ports(1):* Specifies the stimulus port. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	For each parameter, you must specify a different port number. If you specify the same port number for 2 or more parameters, an error occurs and the command is ignored.

Examples (1)

```
SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.THRU = Array(2, 1)
```

Examples (2)

```
Dim ThruPort(1) As Variant
ThruPort(0) = 2
ThruPort(1) = 1
SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.THRU = ThruPort
```

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod[:RESPonse]:THRU <numeric 1>,<numeric 2>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:THRU 2,1"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.SOLT1

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.SOLT1 = *Port*

Description

This command sets the calibration type to the full 1-port calibration of the specified port, when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.DATA command, for the selected channel (*Ch*).

Variable

See *Port*.

Examples

SCPI.SENSe(1).CORRection.COEFficient.METHod.SLOT1 = 1

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod:SOLT1 <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:METH:SOLT1 1"
```

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.SOLT2

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.METHod.SOLT2 = *Ports*

Description

This command sets the calibration type to full 2-port calibration between the two specified ports, when the calibration coefficient data array is written with the SCPI.SENSe(Ch).CORRection.COEFficient.DATA command for the selected channel (*Ch*).

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies a port for full 2-port calibration. • *Ports(1):* Specifies the other port for full 2-port calibration. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	For each parameter, you must specify a different port number. If you specify the same port number for 2 or more parameters, an error occurs and the command is ignored.

Examples (1)

```
SCPI.SENSe(1).CORRection.COEFficient.METHod.SOLT2 = Array(1, 2)
```

Examples (2)

```
Dim CalPort(1) As Variant
CalPort(0) = 1
```

CalPort(1) = 2
SCPI.SENSe(1).CORRection.COEFficient.METHod.SOLT2 = CalPort

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.SAVE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:METHod:SOLT2 <numeric 1>,<numeric 2>

Example of use

10 OUTPUT 717;":SENS1:CORR:COEF:METH:SOLT2 1,2"

SCPI.SENSe(*Ch*).CORRection.COEFficient.SAVE

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COEFficient.SAVE

Description

This command enables the calibration coefficients depending on the selected calibration type from the writing calibration data for the selected channel (*Ch*).

NOTE Enabling the calibration coefficients clears all the calibration data regardless of whether the data are used for the calculation and also clears the calibration type selections.

If you execute this command before all the calibration data needed for calculating the calibration coefficients are written, an error occurs and the command is ignored.

Examples

```
Dim Dmy As Long
Dim Data(3) as Variant
Data(0) = -1.123
Data(1) = 2.456
Data(2) = -2.249
Data(3) = 2.608
SCPI.SENSe(1).CORRection.COEFficient.METHod.RESPonse.THRU = Array(2, 1)
SCPI.SENSe(1).CORRection.COEFficient("ET", 2, 1) = Data
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).CORRection.COEFficient.SAVE
```

Related objects

SCPI.SENSe(Ch).CORRection.COEFficient.DATA

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.ERESponse

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.OPEN

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.SHORt

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.RESPonse.THRU

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT1

SCPI.SENSe(Ch).CORRection.COEFficient.METHod.SOLT2

Equivalent key

Cal > **Calibrate** > **Response (Open)** > **Done**

Cal > **Calibrate** > **Response (Short)** > **Done**

Cal > **Calibrate** > **Response (Thru)** > **Done**

Cal > **Calibrate** > **Enhanced Response** > **Done**

Cal > **Calibrate** > **1-Port Cal** > **Done**

Cal > **Calibrate** > **2-Port Cal** > **Done**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COEFficient:SAVE

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COEF:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.ISOLation

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.ISOLation = *Ports*

Description

This command measures the calibration data of the isolation from the specified stimulus port to the specified response port, for the selected channel (*Ch*).

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies the response port number. • *Ports(1):* Specifies the stimulus port number. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed. If you specify the same port number to 2 port numbers, an error occurs when executed.

Examples

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.ISOLation = Array(1,2)
Dmy = SCPI.IEEE4882.OPC
```

```
Dim IsPort(1) As Variant
Dim Dmy As Long
IsPort(0) = 1
IsPort(1) = 2
```

```
SCPI.SENSe(1).CORRection.COLLect.ACQuire.ISOLation = IsPort
Dmy = SCPI.IEEE4882.OPC
```

Related objects

SCPI.IEEE4882.OPC

Equivalent key

Cal > **Calibrate** > **Response (Thru)** > **Isolation (Optional)**

Cal > **Calibrate** > **Enhanced Response** > **Isolation (Optional)**

Cal > **Calibrate** > **2-Port Cal** > **Isolation (Optional)** > **Port 1-2 Isol**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect[:ACQuire]:ISOLation <numeric 1>,<numeric 2>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ISOL 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.LOAD

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.LOAD = *Port*

Description

This command measures the calibration data of the load standard for the specified port, for the selected channel (*Ch*).

Variable

See *Port*.

Examples

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.LOAD = 1
Dmy = SCPI.IEEE4882.OPC
```

Related objects

SCPI.IEEE4882.OPC

Equivalent key

Cal > **Calibrate** > **Response (Open)**|**Response (Short)** > **Load (Optional)**

Cal > **Calibrate** > **Enhanced Response** > **Load (Optional)**

Cal > **Calibrate** > **1-Port Cal** > **Load**

Cal > **Calibrate** > **2-Port Cal** > **Reflection** > **Port 1 Load**|**Port 2 Load**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect[:ACQuire]:LOAD <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:LOAD 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.OPEN

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.OPEN = *Port*

Description

This command measures the calibration data of the open standard for the specified port, for the selected channel (*Ch*).

Variable

See *Port*.

Examples

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.OPEN = 1
Dmy = SCPI.IEEE4882.OPC
```

Related objects

SCPI.IEEE4882.OPC

Equivalent key

Cal > **Calibrate** > **Response (Open)**|**Enhanced Response**|**1-Port Cal** > **Open**

Cal > **Calibrate** > **2-Port Cal** > **Reflection** > **Port 1 Open**|**Port 2 Open**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect[:ACQuire]:OPEN <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:OPEN 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.SHORt

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.SHORt = *Port*

Description

This command measures the calibration data of the short standard for the specified port, for the selected channel (*Ch*).

Variable

See *Port*.

Examples

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.SHORt = 1
Dmy = SCPI.IEEE4882.OPC
```

Related objects

SCPI.IEEE4882.OPC

Equivalent key

Cal > **Calibrate** > **Response (Short)**|**Enhanced Response**|**1-Port Cal** > **Short**

Cal > **Calibrate** > **2-Port Cal** > **Reflection** > **Port 1 Short**|**Port 2 Short**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect[:ACQuire]:SHORt <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:SHOR 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.THRU

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ACQuire.THRU = *Ports*

Description

This command measures the calibration data of the Thru standard from the specified stimulus port to the specified response port, for the selected channel (*Ch*).

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies the response port. • *Ports(1):* Specifies the stimulus port. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	For each parameter, you must specify a different port number. If you specify the same port number for 2 or more parameters, an error occurs and the command is ignored.

Examples

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.ACQuire.THRU = Array(2,1)
Dmy = SCPI.IEEE4882.OPC
```

```
Dim ThruPort(1) As Variant
Dim Dmy As Long
ThruPort(0) = 2
```

```
ThruPort(1) = 1
SCPI.SENSe(1).CORRection.COLLect.ACQuire.THRU = ThruPort
Dmy = SCPI.IEEE4882.OPC
```

Related objects

SCPI.IEEE4882.OPC

Equivalent key

Cal > **Calibrate** > **Response (Thru)|Enhanced Response** > **Thru**

Cal > **Calibrate** > **2-Port Cal** > **Transmission** > **Port 1-2 Thru**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect[:ACQuire]:THRU <numeric 1>,<numeric 2>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:THRU 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ADAPter(*Pt*).LENGth

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ADAPter(*Pt*).LENGth=*length*

length=SCPI.SENSe(*Ch*).CORRection.COLLect.ADAPter(*Pt*).LENGth

Description

This command sets/displays the approximate length of the adaptor, for the selected channel (*Ch*) and for the selected port.

Variable

Parameter	*length*
Description	Adapter Length
Data type	Double precision floating point type (Double)
Range	-10 to +10
Unit	Second
Note	Adapter length is positive for adaptor removal and negative for adaptor insertion.

Examples

SCPI.SENSe(1).CORRection.COLLect.ADAPter(2).LENGth = 0.01

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ADAPter.REMoval

Equivalent key

Cal > **Calibrate** > **Adapter Removal** > **Port1**|**Port2** > **Coaxial Length**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ADAPter{[1]-2}:LENGth <+ or -><value of length>

Query response

{numeric}<newline><^END>

Example of use

1630

```
10 OUTPUT 717;":SENS1:CORR:COLL:ADAP2:LENG 0.01"
20 OUTPUT 717;":SENS1:CORR:COLL:ADAP2:LENG?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ADAPter(*Pt*).ROTate

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ADAPter(*Pt*).ROTate

Description

This command executes Adapter Removal/Insertion along with moving the phase of adapter (which is removed or inserted) to 180 degrees. This command is useful in cases where auto judgement of phase fails. This command can be executed several times while the calibration remains valid.

NOTE If user cannot execute this command, then "Execution error" is displayed.

Examples

SCPI.SENSe(Ch).CORRection.COLLect.ADAPter(Pt).ROTate

Related Object

SCPI.SENSe(Ch).CORRection.COLLect.ADAPter(Pt).WAVeguide.CUToff

SCPI.SENSe(Ch).CORRection.COLLect.ADAPter(Pt).WAVeguide.LENGth

Equivalent key

Cal > **Calibrate** > **Adapter Removal** > **Port1**|**Port2** > **Rotate Adapter**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ADAPter{[1]-2}:ROTate

Example of use

```
10 OUTPUT 717;"SENS1:CORR:COLL:ADAP2:ROT"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.LOAD

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.LOAD = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.LOAD

Description

This command sets/gets the standard number used for the load measurement of the gain phase port, for a calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim StanLoad as Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.GPORder.LOAD = 3
StanLoad = SCPI.SENSe(1).CORRection.COLLect.CKIT.GPORder.LOAD
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.THRU

Equivalent Key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Load** > **GP Port**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:LOAD <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:LOAD?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:LOAD 3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:LOAD?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.OPEN

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.OPEN = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.OPEN

Description

This command sets/gets the standard number used for the open measurement of the gain phase port, for a calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim StanOpen as Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.GPORder.OPEN = 2
StanOpen = SCPI.SENSe(1).CORRection.COLLect.CKIT.GPORder.OPEN
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.THRU

Equivalent Key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Open** > **GP Port**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:OPEN <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:OPEN?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:OPEN 2"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:OPEN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.SHORt

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.SHORt = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.SHORt

Description

This command sets/gets the standard number used for the short measurement of the gain phase port, for a calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim StanShort as Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.GPORder.SHORt = 1
StanShort = SCPI.SENSe(1).CORRection.COLLect.CKIT.GPORder.SHORt
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.THRU

Equivalent Key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Short** > **GP Port**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:SHORt <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:SHORt?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:SHOR 1"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:SHOR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.THRU

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.THRU = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.GPORder.THRU

Description

This command sets/gets the standard number used for the thru measurement of the gain phase ports, for a calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim StanThru as Long
SCPI.SENSe(2).CORRection.COLLect.CKIT.GPORder.THRU = 4
StanThru = SCPI.SENSe(2).CORRection.COLLect.CKIT.GPORder.THRU
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.GPORder.SHORt

Equivalent Key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Thru** > **GP Port**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:THRU <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:GPORder:THRU?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:THRU 4"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:GPOR:THRU?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.LABel

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.LABel = *Lbl*

Lbl = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.LABel

Description

This command sets/gets the calibration kit name for the calibration kit selected for the selected channels (*Ch*).

Variable

Parameter	*Lbl*
Description	Calibration kit name
Data type	Character string type (String)
Range	254 characters or less
Preset value	Varies depending on the calibration kit number:

Examples

```
Dim CalLbl As String
SCPI.SENSe(1).CORRection.COLLect.CKIT.LABel = "User 1"
CalLbl = SCPI.SENSe(1).CORRection.COLLect.CKIT.LABel
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Label Kit**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:LABel <string>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:LABel?

Query response

{string}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:LAB ""MY_KIT"""
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:LAB?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.LOAD(*Cpt*)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.LOAD(*Cpt*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.LOAD(*Cpt*)

Description

This command sets/gets the standard used for the load measurement of the specified port (*Cpt*), for a calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim StanLoad As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.LOAD(1) = 10
StanLoad = SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.LOAD(1)
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_n)

Equivalent key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Load** > **Set All**|**Port 1**|**Port 2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:LOAD <numeric 1>,<numeric 2>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:LOAD? <numeric 1>

Query response

{numeric 2}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:LOAD 1,9"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:LOAD? 1"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.OPEN(*Cpt*)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.OPEN(*Cpt*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.OPEN(*Cpt*)

Description

This command sets/gets the standard used for the open measurement of the specified port (*Cpt*), for a calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

NOTE Since the variable (*Cpt*) has no preset value, you cannot omit it. If you omit the variable (*Cpt*), an error occurs when executed.

Examples

```
Dim StanOpen As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.OPEN(1) = 10
StanOpen = SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.OPEN(1)
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_n)

Equivalent key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Open** > **Set All**|**Port 1**|**Port 2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:OPEN <numeric 1>,<numeric 2>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:OPEN? <numeric 1>

Query response

{numeric 2}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:OPEN 1,2"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:OPEN? 1"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SHORt(*Cpt*)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SHORt(*Cpt*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.SHORt(*Cpt*)

Description

This command sets/gets the standard used for the short measurement of the specified port (*Cpt*), for the calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim StanShor As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.SHORt(1) = 10
StanShor = SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.SHORt(1)
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_n)

Equivalent key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Short** > **Set All**|**Port 1**|**Port 2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:SHORt <numeric 1>,<numeric 2>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:SHORt? <numeric 1>

Query response

{numeric 2}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:SHOR 1,1"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:SHOR? 1"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.THRU(*Cpt_m,Cpt_n*)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.THRU(*Cpt_m*,*Cpt_n*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ORDer.THRU(*Cpt_m*,*Cpt_n*)

Description

This set/get the standard used for the thru measurement between the specified 2 ports (*Cpt_m* and *Cpt_n*), for the calibration kit selected for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Standard number
Data type	Long integer type (Long)
Range	0 to 21
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim StanThru As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.THRU(1,2) = 10
StanThru = SCPI.SENSe(1).CORRection.COLLect.CKIT.ORDer.THRU(1,2)
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

Equivalent key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Thru** > **Set All**|**Port 1-2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:THRU <numeric 1>,<numeric 2>, <numeric 3>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ORDer:THRU? <numeric 1>,<numeric 2>

Query response

<numeric 3><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:THRU 1,2,11"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ORD:THRU? 1,2"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.RESet

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.RESet

Description

This command resets the calibration kit selected for selected channel (*Ch*) to the default factory setting state.

Examples

SCPI.SENSe(1).CORRection.COLLect.CKIT.RESet

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Restore Cal Kit** > **OK**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:RESet

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:RES"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.SELect

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.SELect = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.SELect

Description

This command sets/gets the calibration kit of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of calibration kit
Data type	Long integer type (Long)
Range	1 to 10
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
Dim CalKit As Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.SELect = 3
CalKit = SCPI.SENSe(1).CORRection.COLLect.CKIT.SELect
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.LOAD(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.OPEN(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.SHORt(Cpt)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ORDer.THRU(Cpt_m,Cpt_n)

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).DELay

Equivalent key

Cal > **Cal Kit**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT[:SELect] <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT[:SELect]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT 3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).ARBitrary

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).ARBitrary = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).ARBitrary

Description

This command sets/gets the value of the arbitrary impedance of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	Value of arbitrary impedance
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	Ω
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanArbt As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).ARBitrary = 50.5
StanArbt = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).ARBitrary
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name**> **Arb. Impedance**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:ARBitrary <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:ARBitrary?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:ARB 50.5"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:ARB?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C0

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C0 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C0

Description

This command sets/gets the value of the CO value of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	C0
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	fF (femto farad): 1E-15 F (farad)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanC0 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C0 = 12.3
StanC0 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C0
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **C0**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C0 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C0?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C0 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C0?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C1

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C1 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C1

Description

This command sets/gets the value of the C1 value of the selected standard (*Std*), for the selected calibration kit and channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	C1
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	1E-27 F/Hz (1E-27 farad / hertz)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanC1 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C1 = 12.3
StanC1 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C1
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name**> **C1**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C1 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C1?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C1 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C2

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C2 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C2

Description

This command sets/gets the value of the C2 value of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	C2
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	1E-36 F/Hz^2 (1E-36 farad /hertz^2)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanC2 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C2 = 12.3
StanC2 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C2
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **C2**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C2 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C2?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C2 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C2?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C3

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C3 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C3

Description

This command sets/gets the value of the C3 value of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	C3
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	1E-45 F/Hz^3 (1E-45 farad / hertz^3)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanC3 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C3 = 12.3
StanC3 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).C3
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **C3**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C3 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:C3?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C3 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:C3?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).CP

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).CP = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).CP

Description

This command sets/gets Cp of the selected standard (*Std*) for the selected channel (*Ch*). This value is used when the model of standard is selected at Equivalent Circuit in SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel. Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

Variable

Parameter	*Value*
Description	Cp
Data Type	Double precision floating point type (Double)
Range	-1E to 1E
Preset Value	-
Unit	F
Resolution	-

Examples

```
Dim Var as Double
Var= 1e-10
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).CP = Var
Var = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).CP
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).LS

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).RS

Equivalent Key

Calibration > Modify Cal Kit > Define STDs > no. name > Cp

Equivalent SCPI Command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:CP <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:CP?
```

Query Response

```
<numeric><newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:CP 1E-10"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:CP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).DELay

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).DELay = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).DELay

Description

This command sets/gets the value of the offset delay of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	Offset delay
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanDel As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).DELay = 12.3
StanDel = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).DELay
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Offset Delay**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:DELay
<numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:DELay?
```

Query response

```
<numeric><newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:DEL 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:DEL?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L0

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L0 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L0

Description

This command sets/gets the value of the L0 value of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	L0
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	pH (pico henry)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanL0 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L0 = 12.3
StanL0 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L0
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L0**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L0 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L0?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L0 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L0?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L1

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L1 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L1

Description

This command sets/gets the value of the L1 value of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	L1
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	1E-24 H/Hz (1E-24 henry / hertz)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanL1 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L1 = 12.3
StanL1 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L1
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L1**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L1 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L1?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L1 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L2

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L2 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L2

Description

This command sets/gets the value of the L2 value of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	L2
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	1E-33 H/Hz^2 (1E-33 henry / $hertz^2$)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanL2 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L2 = 12.3
StanL2 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L2
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L2**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L2 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L2?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L2 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L2?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L3

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L3 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L3

Description

This command sets/gets the value of the L3 value of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	L3
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	1E-42 H/Hz^3 (1E-42 henry / hertz^3)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanL3 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L3 = 12.3
StanL3 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).L3
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **L3**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L3 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:L3?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L3 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:L3?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LS = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LS

Description

This command sets/gets Ls of the selected standard (*Std*) for the selected channel (*Ch*). This value is used when the model of standard is selected at Equivalent Circuit in SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel. Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

Variable

Parameter	*Value*
Description	Ls
Data Type	Double precision floating point type (Double)
Range	-1E to 1E
Preset Value	-
Unit	H
Resolution	-

Examples

```
Dim Var as Double
Var= 1e-10
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).LS = Var
Var = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).LS
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).CP

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).RS

Equivalent Key

Calibration > Modify Cal Kit > Define STDs > no. name > Ls

Equivalent SCPI Command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:LS <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:LS?
```

Query Response

```
<numeric><newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LS 1E-10"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LABel

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LABel = *Lbl*

Lbl = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LABel

Description

This command sets/gets the name of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Lbl*
Description	Standard name
Data type	Character string type (String)
Range	254 characters or less
Preset value	Varies depending on the specified calibration kit and standard.

Examples

```
Dim StanLbl As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LABel = "OPEN 3.5mm"
StanLbl = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LABel
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Label**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:LABel <string>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:LABel?

Query response

{string}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LAB ""OPEN"""
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LAB?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LOSS

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LOSS = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).LOSS

Description

This command sets/gets the value of the offset loss of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	Offset loss
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	Ω/s
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanLoss As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LOSS = 12.3
StanLoss = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).LOSS
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > Modify Cal Kit > Define STDs > no. name > Offset Loss

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:LOSS <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:LOSS?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LOSS 12.3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:LOSS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).MODel

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).MODel = *Model*

Model = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).MODel

Description

This command specify the standard model for the selected standard (*Std*) and for the selected channel (*Ch*). (See Defining Calibration Kit for the model) Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

When you select either equivalent circuit or table, one of either Open, Short, Load and Arbitrary should be selected in SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).TYPE.

Variable

Parameter	*Model*
Description	Standard model of standard kit
Data Type	Character string type (String)
Range	POLYnomial: Polynomial model EQUivalent: Equivalent Circuit model TABLe: Table
Preset Value	-

Examples

```
Dim Model as String
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).MODel = "POLYnomial"
Model = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).MODel
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent Key

Calibration > **Modify Cal Kit** > **Define STDs** > **<standard>** > **STD Model**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:MODel {POLYnomial|EQUivalent|TABLe}

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:MODel?

Query Response

{POLY|EQU|TABL}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:MOD POLY"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:MOD ?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).RS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).RS = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).RS

Description

This command sets/gets Rs of the selected standard (*Std*) for the selected channel (*Ch*). This value is used when the model of standard is selected at Equivalent Circuit in SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel. Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

Variable

Parameter	*Value*
Description	Rs
Data Type	Double precision floating point type (Double)
Range	-1E to 1E
Preset Value	-
Unit	Ω
Resolution	-

Examples

```
Dim Var as Double
Var= 50
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).RS = Var
Var = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).RS
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).CP

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).LS

Equivalent Key

Calibration > **Modify Cal Kit** > **Define STDs** > **no. name** > **Rs**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:RS <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:RS?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:RS 50"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:RS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(Std).TABLe

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(Std).TABLe = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(Std).TABLe

Description

This command sets and gets the the data of table standard model for the selected standard (*Std*) and for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

To select the table model, use the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel and one of either Open, Short, Load and Arbitrary should be selected in SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).TYPE.

Variable

Parameter	*Array*
Description	Indicates the array data of 1 + Num (number of calibration data points) × 3. Where n is an integer between 1 and Num. • *Data(0)* :The number of calibration kit data points you want to set. Specify an integer ranging 1 to 1601. • *Data(n×3-2)* :Frequency at the n-th of standard data. • *Data(n×3-1)* :Real part of data (complex number) at the n-th standard data point. • *Data(n×3)* :Imaginary part of data (complex number) at the n-th standard data point. The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim SetTableAry As Variant, GetTableAray As Variant, NoOfData As Long
```

```
SetTableAry = Array(3, 1000000#, 50.1, 0, 2000000#, 50.05, 0, 3000000#, 51, 0)
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).TABLe = SetTableAry

NoOfData = SCPI.SERVice.CHANnel.STAN.TABLe

ReDim GetTableAray(NoOfData - 1)
GetTableAray = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(1).TABLe
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).MODel

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).TYPE

SCPI.SERVice.CHANnel(Ch).STAN(Std).TABLe

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:TABLe <numeric 1>, <numeric *n*×3-*2*>, <numeric *n*×3-*1*>, <numeric *n*×3> ,..., <numeric Num×3>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:TABLe?

Query response

<numeric 1>, <numeric *n*×3-*2*>, <numeric *n*×3-*1*>, <numeric *n*×3> ,..., <numeric Num×3><newline><^END>

	Description
<numeric 1>	The number of calibration kit data points you want to set. Specify an integer ranging 1 to 4804.
<numeric *n*×3-*2*>	Frequency at the n-th of standard data.
<numeric *n*×3-*1*>	Real part of data (complex number) at the n-th standard data point.
<numeric	Imaginary part of data (complex number) at the n-th

n×3>	standard data point.

Example of use

```
10 Dim A(10)
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:TABL 3, 1000000#, 50.1, 0, 2000000#, 50.05, 0,
3000000#, 51, 0
30 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:TABL?"
40 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).TYPE

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).TYPE = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).TYPE

Description

This command sets/gets the standard type of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Param*
Description	Standard type
Data type	Character string type (String)
Range	Select from either of the following: • "OPEN": Specifies open. • "SHORt": Specifies short. • "LOAD": Specifies load. • "THRU": Specifies Thru. • "UTHRu": Specifies Unknown Thru. • "ARBI": Specifies arbitrary impedance. • "NONE": Specifies DUT of which theoretical value is 0.
Preset value	Varies depending on the specified calibration kit and standard.

Examples

```
Dim StanType As String
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).TYPE = "OPEN"
StanType = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).TYPE
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **STD Type** > **Open|Short|Load|Delay/Thru|Arbitrary|None|Unknown Thru**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:TYPE {OPEN|SHORt|LOAD|THRU|UTHRu|ARBI|NONE}

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:TYPE?

Query response

{OPEN|SHOR|LOAD|THRU|UTHR|ARBI|NONE}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:TYPE OPEN"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:TYPE?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).Z0

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).Z0 = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).Z0

Description

This command sets/gets the value of the offset Z0 of the selected standard (*Std*), for the calibration kit selected for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect.

Variable

Parameter	*Value*
Description	Offset Z0
Data type	Double precision floating point type (Double)
Range	-1E18 to 1E18
Preset value	Varies depending on the specified calibration kit and standard.
Unit	Ω
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StanZ0 As Double
SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).Z0 = 50
StanZ0 = SCPI.SENSe(1).CORRection.COLLect.CKIT.STAN(5).Z0
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

Equivalent key

Cal > **Modify Cal Kit** > **Define STDs** > **no. name** > **Offset Z0**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:Z0 <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:STAN{[1]-21}:Z0?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:Z0 50"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:STAN1:Z0?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.LOAD

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.LOAD = *Std*

Std = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.LOAD

Description

This command sets/gets standard number for load in impedance calibration for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

Variable

Parameter	*Std*
Description	standard number for load
Data Type	Long integer type (Long)
Range	1 ~ 21
Preset Value	3 (for registered calibration kit)
Unit	-
Resolution	-

Examples

```
Dim Var as Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ZORDer.LOAD = 3
Var = SCPI.SENSe(1).CORRection.COLLect.CKIT.ZORDer.LOAD
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ZORDer.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ZORDer.SHORt

Equivalent Key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Load** > **Impedance Cal**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ZORDer:LOAD <numeric>
:SENSe{[1]-4}:CORRection:COLLect:CKIT:ZORDer:LOAD?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ZORD:LOAD 3"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ZORD:LOAD?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.OPEN

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.OPEN = *Std*

Std = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.OPEN

Description

This command sets/gets standard number for open in impedance calibration for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

Variable

Parameter	*Std*
Description	Standard number for open
Data Type	Long integer type (Long)
Range	1 ~ 21
Preset Value	1 (for registered calibration kit)
Unit	-
Resolution	-

Examples

```
Dim Var as Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ZORDer.OPEN = 1
Var = SCPI.SENSe(1).CORRection.COLLect.CKIT.ZORDer.OPEN
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ZORDer.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ZORDer.SHORt

Equivalent Key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Open** > **Impedance Cal**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ZORDer:OPEN <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ZORDer:OPEN?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ZORD:OPEN 1"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ZORD:OPEN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.SHORt

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.SHORt = *Std*

Std = SCPI.SENSe(*Ch*).CORRection.COLLect.CKIT.ZORDer.SHORt

Description

This command sets/gets standard number for short in impedance calibration for the selected channel (*Ch*). Before defining this, the calibration kit should be selected kit by the SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect. This command is available when option 005 is installed.

Variable

Parameter	*Std*
Description	Standard number for short
Data Type	Long integer type (Long)
Range	1 ~ 21
Preset Value	2 (for registered calibration kit)
Unit	-
Resolution	-

Examples

```
Dim Var as Long
SCPI.SENSe(1).CORRection.COLLect.CKIT.ZORDer.SHORt = 2
Var = SCPI.SENSe(1).CORRection.COLLect.CKIT.ZORDer.SHORt
```

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.SELect

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ZORDer.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.ZORDer.LOAD

Equivalent Key

Cal > **Modify Cal Kit** > **Specify CLSs** > **Short** > **Impedance Cal**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ZORDer:SHORt <numeric>

:SENSe{[1]-4}:CORRection:COLLect:CKIT:ZORDer:SHORt?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ZORD:SHOR 2"
20 OUTPUT 717;":SENS1:CORR:COLL:CKIT:ZORD:SHOR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.CLEar

Type of object

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.CLEar

Description

This command clears the calibration measurement data when the frequency offset feature is off, for the selected channel (*Ch*).

NOTE Settings that have been temporarily changed due to measurement for each standard (number of traces, measurement parameters, and so on) return to their original values.

Example of use

SCPI.SENSe(1).CORRection.COLLect.CLEar

Related objects

SCPI.SENSe(Ch).OFFSet.STATe

Equivalent key

Cal > **Calibrate** > **Responce(Open)** > **Cancel** > **OK**

Cal > **Calibrate** > **Responce(Short)** > **Cancel** > **OK**

Cal > **Calibrate** > **Responce(Thru)** > **Cancel** > **OK**

Cal > **Calibrate** > **Enhanced Responce** > **Cancel** > **OK**

Cal> **Calibrate** > **1-Port Cal** > **Cancel** > **OK**

Cal> **Calibrate** > **2-Port Cal** > **Cancel** > **OK**

Cal > **Calibrate** > **Adapter Removal** > **Port1**|**Port2** > **Cancel** > **OK**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:CLEar

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:CLE"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.CCHeck.ACQuire

Type of object

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.CCHeck.ACQuire

Description

This command executes the confidence check of the calibration coefficients for selected channel (*Ch*), using ECal (Electronic Calibration). In other words, this command sets the data measured with the analyzer and the data stored in ECal so that they can be compared).

NOTE If you execute this object when the ECal module is not connected or when ports are not connected each other appropriately, a runtime error occurs.

Example of use

SCPI.SENSe(1).CORRection.COLLect.ECAL.CCHeck.ACQuire

Equivalent key

Cal > **ECal** > **Confidence Check**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ECAL:CCHeck[:ACQuire]

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:CCH"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ERESponse

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ERESponse = *Eports*

Description

This command executes enhanced response calibration between the two specified ports of the selected channel using the ECal (Electrical Calibration) module.

NOTE If you execute this command when the ECal module is not connected or when ports are not connected each other appropriately, an error occurs and the command is ignored.

Variable

Parameter	*Eports*
Description	Indicates 2-element array data (port number). • *EPorts(0):* Specifies the response port. • *EPorts(1):* Specifies the stimulus port. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	For each parameter, you must specify a different port number. If you specify the same port number for 2 or more parameters, an error occurs and the command is ignored.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim ERESPort(1) As Variant
ERESPort(0) = 1
ERESPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ECAL.ERESponse = ERESPort
```

Equivalent key

Cal > ECal > Enhanced Response > 1-2 (S12 S22)|2-1 (S21 S11)

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ECAL:ERESponse <numeric 1>,<numeric 2>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ERES 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.INFormation

Object type

Property (**Read Only**)

Syntax

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.INFormation?

Description

This command gets information of the connected ECAL modules to the E5061B.

Variable

The command return information in a string variable <param> in the following syntax:

[For 2-port ECal]

- ModelNumber
- SerialNumber
- PortAConnector
- PortBConnector
- MinFreq
- MaxFreq
- NumberOfPoints
- Calibrated
- ID
- PortAExtension
- PortBExtension
- Analyzer
- Operator
- Location

[For 4-port ECal]

- ModelNumber
- SerialNumber
- PortAConnector
- PortBConnector
- PortCConnector
- PortDConnector
- MinFreq
- MaxFreq

- NumberOfPoints
- Calibrated
- ID
- PortAExtension
- PortBExtension
- PortCExtension
- PortDExtension
- Analyzer
- Operator
- Location

Examples

```
Dim Cal_Info As String
Cal_Info = SCPI.SENS1.CORRection.COLLect.ECAL.INFormation
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.SAVE

SCPI.SENSe(Ch).CORRection.TYPE(Tr)

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ECAL:INFormation?

Query response

{Model number,Serial number,Connector type, Calibration date, Min and max frequency}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:INF?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ISOLation.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ISOLation.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ISOLation.STATe

Description

This command ignores ON/OFF of the isolation measurement when executing Ecal for the selected channel (*Ch*), because the isolation of the E5061B is better than that of the ECal. This command takes no action and only exists to maintain backward compatibility.

Variable

Parameter	*Status*
Description	ON/OFF of the isolation measurement when executing ECal
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the isolation measurement. • False or OFF: Turns OFF the isolation measurement.
Preset value	False or OFF

Examples

```
Dim EcalIso As Boolean
SCPI.SENSe1.CORRection.COLLect.ECAL.ISOLation.STATe = False
EcalIso = SCPI.SENSe1.CORRection.COLLect.ECAL.ISOLation.STATe
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT1

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.SOLT2

Equivalent key

No equivalent key is available on front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ECAL:ISOLation[:STATe] {ON|OFF|1|0}

:SENSe{[1]-4}:CORRection:COLLect:ECAL:ISOLation[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ISOL OFF"
20 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ISOL?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ORIentation.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ORIentation.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.ORIentation.STATe

Description

This command turns ON/OFF the ECal auto detect function.

Variable

Parameter	*Status*
Description	ON/OFF the ECal auto detect funcion.
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the auto detect function. • False or OFF: Turns OFF the auto detect function.
Preset value	True or ON

For information on the variable (*Ch*), see *Ch*.

Examples

```
Dim EcalOri As Boolean
SCPI.SENSe(1).CORRection.COLLect.ECAL.ORIentation.STATe = True
EcalOri = SCPI.SENSe(1).CORRection.COLLect.ECAL.ORIentation.STATe
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(Cpt)

Equivalent key

Cal > **ECal** > **Orientation** > **Orientation**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ECAL:ORIentation[:STATe] {ON|OFF|1|0}

:SENSe{[1]-4}:CORRection:COLLect:ECAL:ORIentation[:STATe]?

Query response

{0|1}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ORI ON"
20 OUTPUT 717;":SENS1:CORR:COLL:ECAL:ORI?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.PATH(*Cpt*)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(*Cpt*) = *Value*

Value = SCPI.SENSe(Ch).CORRection.COLLect.ECAL.PATH(*Cpt*)

Description

This command sets/gets the ECal module n port number which is connected to a specified port.

Variable

Parameter	*Value*
Description	Port of ECal module.
Data type	Long integer type (Long)
Range	One of the following is set/get: • 0: Nothing is connected. • 1: Port A is connected. • 2: Port B is connected. • 3: Port C is connected. • 4: Port D is connected.

For information on the variable (*Cpt*), see *Cpt*.

Examples

```
Dim ECalPort As Long
SCPI.SENSe.CORRection.COLLect.ECAL.PATH(1) = 3
ECalPort = SCPI.SENSe.CORRection.COLLect.ECAL.PATH(1)
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.ECAL.ORIentation.STATe

Equivalent key

Cal > **ECal** > **Orientation** > **Port1**|**Port2** > **A**|**B**|**C**|**D**|**None**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:ECAL:PATH <numeric 1>,<numeric 2>

:SENSe{[1]-4}:CORRection:COLLect:ECAL:PATH? <numeric 2>

Example of use

10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:PATH 1,2"

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT1

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT1 = *Eport*

Description

This command executes 1-port calibration of the specified port of selected channel (*Ch*) using the ECal (Electronic Calibration) module.

NOTE If this command is executed when the ECal module is not connected or when ports are not connected each other appropriately, an error occurs and the command is ignored.

Variable

Parameter	*Eport*
Description	Port number
Data type	Long integer type (Long)
Range	1 to 2
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Examples

```
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT1 = 1
```

Equivalent key

Cal > **ECal** > **1-Port Cal** > **Port 1**|**Port 2**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:ECAL:SOLT1 <numeric>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:SOLT1 1"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT2

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.SOLT2 = *Eports*

Description

Executes full 2-port calibration between the specified 2 ports of channels 1 to 16 (*Ch*) using the ECal (Electronic Calibration) module.

NOTE If this command is executed when the ECal module is not connected or when ports are not connected each other appropriately, an error occurs and the command is ignored.

Variable

Parameter	*Eports*
Description	Indicates 2-element array data (port number). • *EPorts(0)* *EPorts*(1) Specifies the port numbers for 2-port ECal. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed. If you specify the same port number to 2 port numbers, an error occurs when executed. The order of the 2 port numbers to be specified is arbitrary.

Examples

SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT2 = Array(1,2)

```
Dim EcalPort(1) As Variant
EcalPort(0) = 1
EcalPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ECAL.SOLT2 = EcalPort
```

Equivalent key

Cal > **ECal** > **2-Port Cal**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:ECAL:SOLT2 <numeric 1>,<numeric 2>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:SOLT2 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.THRU

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.THRU = *Eports*

Description

This command executes response calibration (thru) between the specified 2 ports of the selected channel (*Ch*) using the ECal (Electronic Calibration) module.

If this command is executed when the ECal module is not connected or when ports are not connected to each other appropriately, an error occurs and the command is ignored.

Variable

Parameter	*Eports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies the response port number. • *Ports(1):* Specifies the stimulus port number. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 4
Resolution	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed. If you specify the same port number to 2 port numbers, an error occurs when executed. The order of the 2 port numbers to be specified is arbitrary.

Examples

```
SCPI.SENSe(1).CORRection.COLLect.ECAL.THRU = Array(1,2)

Dim EcalPort(1) As Variant
EcalPort(0) = 1
```

```
EcalPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.ECAL.THRU = EcalPort
```

Equivalent key

Cal > ECal > Thru Cal > 1-2 (S12)|2-1 (S21)

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:ECAL:THRU <numeric 1>,<numeric 2>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:THRU 1,2"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.UCHar

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.ECAL.UCHar = *Param*

Param = *S*CPI.SENSe(*Ch*).CORRection.COLLect.ECAL.UCHar

Description

This command sets the ECal characteristic used when executing the user-defined ECal, for the selected channel (*Ch*).

NOTE The user-defined ECal is a type of ECal that is executed using the characteristic that has been acquired by the user and stored in the memory for ECal.

When the ECal module is not connected or the characteristic is not stored at the specified location number, executing this object causes a runtime error.

Variable

Parameter	*Param*
Description	Characteristic used when executing ECal (user characterization)
Data type	Character string type (String)
Range	Select from either of the following: • "CHAR0": Uses the factory-default characteristic. (Normal ECal) • "CHAR1": Uses the characteristic stored at location number 1 in the ECal's flash memory. • "CHAR2": Uses the characteristic stored at location number 2 in the ECal's flash memory. • "CHAR3": Uses the characteristic stored at location number 3 in the ECal's flash memory. • "CHAR4": Uses the characteristic stored at location number 4 in the ECal's flash memory. • "CHAR5": Uses the characteristic stored at

	location number 5 in the ECal's flash memory.
Preset value	"CHAR0"

Examples

```
Dim UserChar As String
SCPI.SENSe(1).CORRection.COLLect.ECAL.UCHar = "CHAR2"
UserChar = SCPI.SENSe(1).CORRection.COLLect.ECAL.UCHar
```

Equivalent key

Cal > **ECal** > **Characterization** > **Factory**|**User1**|**User2**|**User3**|**User4**|**User5**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:ECAL:UCHar
{CHAR0|CHAR1|CHAR2|CHAR3|CHAR4|CHAR5}
:SENSe{[1]-4}:CORRection:COLLect:ECAL:UCHar?
```

Query response

```
{CHAR0|CHAR1|CHAR2|CHAR3|CHAR4|CHAR5}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:ECAL:UCH CHAR1"
20 OUTPUT 717;":SENS1:CORR:COLL:ECAL:UCH?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.ISOLation

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.ISOLation

Description

This command measures the calibration data of the isolation from the T-port to the R-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.GPACquire.ISOLation

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.THRU

Equivalent Key

Cal > **Calibrate** > **Response (Thru)** > **Isolation (Optional)**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:GPACquire:ISOLation

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:GPAC:ISOL"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.LOAD

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.LOAD

Description

This command measures the calibration data of the load standard for the T-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(2).CORRection.COLLect.GPACquire.LOAD

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.ISOLation

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.THRU

Equivalent Key

Cal > **Calibrate** > **Response (Open)** > **Load (Optional)**

Cal > **Calibrate** > **Response (Short)** > **Load (Optional)**

Cal > **Calibrate** > **1-Port Cal** > **Load**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:GPACquire:LOAD

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:GPAC:LOAD"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.OPEN

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.OPEN

Description

This command measures the calibration data of the open standard for the T-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.GPACquire.OPEN

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.ISOLation

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.THRU

Equivalent Key

Cal > **Calibrate** > **Response (Open)** > **Open**

Cal > **Calibrate** > **1-Port Cal** > **Open**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:GPACquire:OPEN

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:GPAC:OPEN"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.SHORt

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.SHORt

Description

This command measures the calibration data of the short standard for the T-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.GPACquire.SHORt

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.ISOLation

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.THRU

Equivalent Key

Cal > **Calibrate** > **Response (Short)** > **Short**

Cal > **Calibrate** > **1-Port Cal** > **Short**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:GPACquire:SHORt

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:GPAC:SHOR"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.THRU

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.GPACquire.THRU

Description

This command measures the calibration data of the thru standard from the T-port to the R-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.GPACquire.THRU

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.ISOLation

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.LOAD

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.GPACquire.SHORt

Equivalent Key

Cal > **Calibrate** > **Response (Thru)** > **Thru**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:GPACquire:THRU

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:GPAC:THRU"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.ADAPter.REMoval

Object type

Property (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.ADAPter.REMoval = *Port*

Description

This command sets the port in which adaptor needs to be added/removed.

Variable

See *Port*.

Examples

SCPI.SENSe(Ch).CORRection.COLLect.METHod.ADAPter.REMoval = 1

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.ADAPter(Pt).LENGth

Equivalent key

Cal > **Calibrate** > **Adapter Removal** > **Port1**|**Port2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:ADAPter:REMoval {1-2}

Example of use

```
Call viVPrintf(vi, ":SENS1:CORR:COLL:METH:ADAP:REM  1"+vbcr, 0)
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.ERESponse

Object type

Property (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.ERESponse = *Ports*

Description

This command sets the calibration type to the enhanced response calibration between the two specified ports, for the selected channel.

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies the response port. • *Ports(1):* Specifies the stimulus port. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	For each parameter, you must specify a different port number. If you specify the same port number for 2 or more parameters, an error occurs and the command is ignored.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim ERESPort(1) As Variant
ERESPort(0) = 1
ERESPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.METHod.ERESponse = ERESPort
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Equivalent key

Cal > **Calibrate** > **Enhanced Response** > **Select Ports** > **1-2 (S12 S22)|2-1 (S21 S11)**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:ERESponse <numeric 1>,<numeric 2>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:ERES 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPResponse.OPEN

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPResponse.OPEN

Description

This command sets the calibration type to the response calibration (open) of the T-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.METHod.GPResponse.OPEN

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.THRU

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPS1

Equivalent Key

Cal > **Calibrate** > **Response (Open)** > **Select Port** > **Gain Phase**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:GPResponse:OPEN

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:GPR:OPEN"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPResponse.SHORt

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPResponse.SHORt

Description

This command sets the calibration type to the response calibration (short) of the T-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.METHod.GPResponse.SHORt

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.THRU

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPS1

Equivalent Key

Cal > **Calibrate** > **Response (Short)** > **Select Port** > **Gain Phase**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:GPResponse:SHORt

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:GPR:SHOR"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPResponse.THRU

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPResponse.THRU

Description

This command sets the calibration type to the response calibration (thru) between the T-port and the R-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.METHod.GPResponse.THRU

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPS1

Equivalent Key

Cal > **Calibrate** > **Response (Thru)** > **Select Ports** > **GP Port**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:GPResponse:THRU

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:GPR:THRU"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPS1

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.GPS1

Description

This command sets the calibration type to the full 1-port SOLT calibration of the T-port using gain phase measurement, for the selected channel (*Ch*).

Examples

SCPI.SENSe(2).CORRection.COLLect.METHod.GPS1

Related Objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.METHod.GPResponse.THRU

Equivalent Key

Cal > **Calibrate** > **1-Port Cal** > **Select Port** > **Gain Phase**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:GPS1

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:GPS1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.OPEN

Object type

Property (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.OPEN = *Port*

Description

This command sets the calibration type to the response calibration (open) of the specified port, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.OPEN = 1

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Equivalent key

Cal > **Calibrate** > **Response (Open)** > **Select Port** > **1|2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod[:RESPonse]:OPEN <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:OPEN 1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.SHORt

Object type

Property (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.SHORt = *Port*

Description

This command sets the calibration type to the response calibration (short) of the specified port, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.SHORt = 1

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Equivalent key

Cal > **Calibrate** > **Response (Short)** > **Select Port** > **1|2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod[:RESPonse]:SHORt <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SHOR 1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.THRU

Object type

Property (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.RESPonse.THRU = *Ports*

Description

This command sets the calibration type to the response calibration (thru) between the specified 2 ports, for the selected channel (*Ch*).

Variable

For information on the variable (*Ch*) and the variable (*Port*), see Ch and Port.

Examples

```
SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.THRU = Array(2,1)
```

```
Dim ThruPort(1) As Variant
ThruPort(0) = 2
ThruPort(1) = 1
SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.THRU = ThruPort
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Equivalent key

Cal > **Calibrate** > **Response (Thru)** > **Select Ports** > **1-2 (S12)**|**2-1 (S21)**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod[:RESPonse]:THRU <numeric 1>,<numeric 2>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:THRU 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT1

Object type

Property (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT1 = *Port*

Description

This command sets the calibration type to the 1-port calibration of the specified port, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT1 = 1

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Equivalent key

Cal > **Calibrate** > **1-Port Cal > Select Port** > **1|2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:SOLT1 <numeric>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SOLT1 1"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT2

Object type

Property (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.SOLT2 = *Ports*

Description

This command sets the calibration type to the full 2-port calibration between the specified 2 ports, for the selected channel (*Ch*).

Variable

Parameter	*Ports*
Description	Indicates 2-element array data (port number). • *Ports(0):* Specifies a port for full 2-port calibration. • *Ports(1):* Specifies the other port for full 2-port calibration. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	1 to 2
Resolution	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed. If you specify the same port number to 2 port numbers, an error occurs when executed. The order of the 2 port numbers to be specified is arbitrary.

Examples

```
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT2 = Array(1,2)
```

```
Dim CalPort(1) As Variant
CalPort(0) = 1
CalPort(1) = 2
SCPI.SENSe(1).CORRection.COLLect.METHod.SOLT2 = CalPort
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.TYPE

Equivalent key

Cal > Calibrate > 2-Port Cal

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:COLLect:METHod:SOLT2 <numeric
1>,<numeric 2>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:SOLT2 1,2"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TYPE

Object type

Property (**Read Only**)

Syntax

Param = SCPI.SENSe(*Ch*).CORRection.COLLect.METHod.TYPE

Description

This command reads the selected calibration type of selected channel (*Ch*).

NOTE This object is used to check the selected calibration type for calculating the calibration coefficients. To check the applied calibration type (error correction on), use the SCPI.SENSe(Ch).CORRection.TYPE(Tr) object.

Variable

Parameter	*Param*
Description	Calibration type
Range	AREM — Adaptor Removal ERES — Enhanced response calibration NONE — None RESPO — Response calibration (open) RESPS — Response calibration (short) RESPT — Response calibration (thru) SOLT1 — 1-port calibration. SOLT2 — Full 2-port calibration.
Preset Value	NONE

Data type	Character string type (String)

Examples

```
Dim CalType As String
CalType = SCPI.SENSe(1).CORRection.COLLect.METHod.TYPE
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.SAVE

SCPI.SENSe(Ch).CORRection.TYPE(Tr)

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:METHod:TYPE?

Query response

{AREM|ERES|NONE|RESPO|RESPS|RESPT|SOLT1|SOLT2}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:METH:TYPE?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.COLLect.PARTial.SAVE

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.PARTial.SAVE

Description

This command, used for partial overwrite, recalculates the calibration coefficients depending on the selected calibration type from the measured calibration data.

NOTE Calculating the calibration coefficients clears all calibration data regardless of whether they are used for the calculation and also clears the calibration type selections.

If partial overwrite is executed before selecting the calibration type, an error occurs and the command is ignored.

Variable

For information on the variable (*Ch*), see Ch.

Examples

SCPI.SENSe(1).CORRection.COLLect.PARTial.SAVE

Equivalent key

Cal > **Calibrate** > **1-Port Cal**|**2-Port Cal** > **Overwrite**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:PARTial:SAVE

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:PART:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.COLLect.SAVE

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.COLLect.SAVE

Description

This command calculates the calibration coefficients depending on the calibration type selection, from the measured calibration data.

NOTE Calculating the calibration coefficients clears all the measured calibration data whether or not used for the calculation and also clears the calibration type selection.

If this command is executed before all necessary calibration data for calculating the calibration coefficients is measured, an error occurs and the command is ignored.

Variable

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Dmy As Long
SCPI.SENSe(1).CORRection.COLLect.METHod.RESPonse.THRU = Array(2,1)
SCPI.SENSe(1).CORRection.COLLect.ACQuire.THRU = Array(2,1)
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).CORRection.COLLect.SAVE
```

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.METHod.RESPonse.OPEN

SCPI.SENSe(Ch).CORRection.COLLect.METHod.RESPonse.SHORt

SCPI.SENSe(Ch).CORRection.COLLect.METHod.RESPonse.THRU

SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT1

SCPI.SENSe(Ch).CORRection.COLLect.METHod.SOLT2

Equivalent key

Cal > **Calibrate** > **Response(XXXX)**|**n-Port Cal**|**Enhanced Response** > **Done**

Cal > **Calibrate** > **Adapter Removal** > **Port1**|**Port2** > **Done**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:COLLect:SAVE

Example of use

```
10 OUTPUT 717;":SENS1:CORR:COLL:SAVE"
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.CONFig

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.CONFig = *Param*

Param = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.CONFig

Description

This command sets/gets the frequency point to calculate the auto port extension, for the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	The frequency point to calculate the auto port extension
Data type	Character string type (String)
Range	Select from either of the following: • "CSPN": Uses the frequency of the current sweep range. • "AMKR": Use the frequency of the active marker. This is applied to Loss 1 and Loss 2 is ignored. • "USPN": This is executed with the arbitrary specified start frequency and stop frequency.
Preset value	"CSPN"

Examples

```
Dim Conf As String
SCPI.SENSe(1).CORRection.EXTension.AUTO.CONFig = "AMKR"
Conf = SCPI.SENSe(1).CORRection.EXTension.AUTO.CONFig
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Equivalent key

Cal > **Port Extensions** > **Auto Port Extension** > **Method** > **Current Span**|**Active Marker**|**User Span**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:AUTO:CONFig {CSPN|AMKR|USPN}

:SENSe{[1]-4}:CORRection:EXTension:AUTO:CONFig?

Query response

{CSPN|AMKR|USPN}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:CONF CSPN"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:CONF?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.DCOFfset

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.DCOFfset = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.DCOFfset

Description

This command enables/disables or gets the usage of DC Offset value for the results of the auto port extension, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the usage of DC Offset value for the results of the auto port extension
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Uses the DC Offset value for the results. • False or OFF: Does not use the DC Offset value for the results.
Preset value	False or OFF

Examples

```
Dim Dcof As Boolean
SCPI.SENSe(1).CORRection.EXTension.AUTO.DCOFfset = True
Dcof = SCPI.SENSe(1).CORRection.EXTension.AUTO.DCOFfset
```

Related objects

SCPI.CALCulate(Ch).PARameter(Tr).SELect

SCPI.CALCulate(Ch).SELected.CONVersion.FUNCtion

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Equivalent key

Cal > **Port Extensions** > **Auto Port Extension** > **Adjust Mismatch**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:AUTO:DCOFfset {ON|OFF|1|0}

:SENSe{[1]-4}:CORRection:EXTension:AUTO:DCOFfset?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:DCOF ON"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:DCOF?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.LOSS

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.LOSS = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.LOSS

Description

This command turns ON/OFF or gets the status of the loss compensation for the results of the auto port extension, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the loss compensation for the results of the auto port extension
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the loss compensation • False or OFF: Turns OFF the loss compensation
Preset value	False or OFF

Examples

```
Dim AutoLoss As Boolean
SCPI.SENSe(1).CORRection.EXTension.AUTO.LOSS = True
AutoLoss = SCPI.SENSe(1).CORRection.EXTension.AUTO.LOSS
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Equivalent key

Cal > **Port Extensions** > **Auto Port Extension** > **Include Loss**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:EXTension:AUTO:LOSS {ON|OFF|1|0}
:SENSe{[1]-4}:CORRection:EXTension:AUTO:LOSS?
```

Query response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:LOSS ON"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:LOSS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.MEASure

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.MEASure = *Param*

Description

This command measures the calibration data of the OPEN standard or SHORT standard of the auto port extension, for the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Standard type of the auto port extension
Data type	Character string type (String)
Range	Select from either of the following: • "OPEN": Measures the calibration data of the OPEN standard • "SHORt": Measures the calibration data of the SHORT standard
Preset value	"SHORt"

Examples

```
Dim AutoMeas As String
SCPI.SENSe(1).CORRection.EXTension.AUTO.MEASure = "OPEN"
AutoLoss = SCPI.SENSe(1).CORRection.EXTension.AUTO.LOSS
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Equivalent key

Cal > Port Extensions > Auto Port Extension > Measure OPEN|Measure Short > All|Port 1|Port 2

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:EXTension:AUTO:MEASure {OPEN|SHORt}
:SENSe{[1]-4}:CORRection:EXTension:AUTO:MEASure?
```

Query response

```
{OPEN|SHOR}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:MEAS OPEN"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:MEAS?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.PORT(*Pt*)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.PORT(*Pt*) = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.PORT(*Pt*)

Description

This command turns ON/OFF or gets the status of the auto port extension, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the auto port extension
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the auto port extension • False or OFF: Turns OFF the auto port extension
Preset value	True or ON

For information on the variable (*Ch*) or Pt, refer to Ch or Pt.

Examples

```
Dim APort As Boolean
SCPI.SENSe(1).CORRection.EXTension.AUTO.PORT(1) = True
APort = SCPI.SENSe(1).CORRection.EXTension.AUTO.PORT(1)
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Equivalent key

Cal > **Port Extensions** > **Auto Port Extension** > **Select Ports** > **Port 1**|**Port 2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:AUTO:PORT{[1]|2} {ON|OFF|1|0}

:SENSe{[1]-4}:CORRection:EXTension:AUTO:PORT{[1]|2}?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:PORT1 ON"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:PORT1?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.RESet

Object type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.RESet

Description

This command deletes the finished measurement data (OPEN and SHORT), for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).CORRection.EXTension.AUTO.RESet

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:AUTO:RESet

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:RES"
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STARt

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STARt = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STARt

Description

This command gets/sets the start frequency within the frequency range of the user specified auto port extension, for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Start frequency
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	5
Unit	Hz (hertz)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim AStart As Double
SCPI.SENSe(1).CORRection.EXTension.AUTO.STARt = 1E9
AStart = SCPI.SENSe(1).CORRection.EXTension.AUTO.STARt
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STOP

Equivalent key

Cal > Port Extensions > Auto Port Extension > Method > User Span Start

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:AUTO:STARt <numeric>
:SENSe{[1]-4}:CORRection:EXTension:AUTO:STARt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STAR 1.2E9"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STAR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STOP

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STOP = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.AUTO.STOP

Description

This command sets/gets the stop frequency within the frequency range of the user specified auto port extension, for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Stop frequency
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	3000000000
Unit	Hz (hertz)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim AStop As Double
SCPI.SENSe(1).CORRection.EXTension.AUTO.STOP = 1E9
AStop = SCPI.SENSe(1).CORRection.EXTension.STOP.STARt
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.CONFig

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet
SCPI.SENSe(Ch).CORRection.EXTension.AUTO.STARt

Equivalent key

Cal > **Port Extensions** > **Auto Port Extension** > **Method** > **User Span Stop**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:AUTO:STOP <numeric>
:SENSe{[1]-4}:CORRection:EXTension:AUTO:STOP?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STOP 1E9"
20 OUTPUT 717;":SENS1:CORR:EXT:AUTO:STOP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).FREQuency(*Fq*)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).FREQuency(*Fq*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).FREQuency(*Fq*)

Description

This command sets/gets the frequency used for calculation of the loss value of the frequency 1 and 2 (*Fq*) of the selected port (*Pt*), for the selected channel (*Ch*).

Variable

Parameter	*Fq*
Description	Frequency number
Data type	Long integer type (Long)
Range	1 to 2
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Parameter	*Value*
Description	Frequency
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	1E9

Unit	Hz (hertz)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PortFreq As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).FREQuency(1) = 500E6
PortFreq = SCPI.SENSe(1).CORRection.EXTension.PORT(1).FREQuency(1)
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)

Equivalent key

Cal > **Port Extensions** > **Extension Port 1**|**Extension Port 2** > **Loss** > **Freq1**|**Freq2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:FREQuency{[1]|2} <numeric>

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:FREQuency{[1]|2}?

Query response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:FREQ1 10E6"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:FREQ1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).INCLude(*Il*).STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).INCLude(*Il*).STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).INCLude(*Il*).STATe

Description

This command turns ON/OFF the set of loss value and frequency value of include 1 and 2 (*Il*) of the port 1 to 2 (*Pt*), for the selected channel (*Ch*).

Variable

Parameter	*Il*
Description	Include number
Data type	Long integer type (Long)
Range	1 to 2
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Parameter	*Status*
Description	ON/OFF the set of loss value and frequency value.
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the loss value and frequency value.

	• False or OFF: Turns OFF the loss value and frequency value.
Preset value	False or OFF

Examples

```
Dim PortIncl As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).INCLude(1).STATe = 500E6
PortIncl = SCPI.SENSe(1).CORRection.EXTension.PORT(1).INCLude(1).STATe
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Equivalent key

Cal > **Port Extensions** > **Extension Port 1**|**Extension Port 2** > **Loss** > **Loss1**|**Loss2**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:INCLude{[1]|2}[:STATe] {ON|OFF|1|0}
:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:INCLude{[1]|2}[:STATe]?
```

Query response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:INCL1 ON"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:INCL1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LDC

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LDC = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LDC

Description

This command sets/gets the DC loss value of the port 1 to 2 (*Pt*), for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	The loss value of DC.
Data type	Double precision floating point type (Double)
Range	-90 to 90
Preset value	0
Unit	dB
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PLdc As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).LDC = 45
PLdc = SCPI.SENSe(1).CORRection.EXTension.PORT(1).LDC
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Equivalent key

Cal > **Port Extensions** > **Extension Port 1**|**Extension Port 2** > **Loss** > **Loss at DC**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:LDC <numeric>

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:LDC?

Query response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LDC 1.2"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LDC?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LOSS(Loss)

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LOSS(*Loss*) = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).LOSS(*Loss*)

Description

This command sets/gets the loss value of the loss 1 to 2 of the port 1 to 2 (*Pt*), for the selected channel (*Ch*).

Variable

Parameter	*Loss*
Description	Loss number
Data type	Long integer type (Long)
Range	1 to 2
Preset value	1
Note	If the specified variable is out of the allowable setup range, an error occurs when executed.

Parameter	*Value*
Description	The loss value
Data type	Double precision floating point type (Double)
Range	-90 to 90
Preset value	0

Unit	dB
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PLoss As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).LOSS(1) = -45
PLoss = SCPI.SENSe(1).CORRection.EXTension.PORT(1).LOSS(1)
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Equivalent key

Cal > **Port Extensions** > **Extension Port 1**|**Extension Port 2** > **Loss** > **Loss1**|**Loss2**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:LOSS{[1]|2} <numeric>

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:LOSS{[1]|2}?

Query response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LOSS1 0.8"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:LOSS1?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).TIME

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).TIME = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.EXTension.PORT(*Pt*).TIME

Description

This command sets/gets the delay time for the port extension of ports 1 and 2 (*Pt*), for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Delay time
Data type	Double precision floating point type (Double)
Range	-10 to 10
Preset value	0
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PortExt As Double
SCPI.SENSe(1).CORRection.EXTension.PORT(1).TIME = 1E-3
PortExt = SCPI.SENSe(1).CORRection.EXTension.PORT(1).TIME
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)

Equivalent key

Cal > **Port Extensions** > **Extension Port 1**|**Extension Port 2** > **Extension**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:TIME <numeric>

:SENSe{[1]-4}:CORRection:EXTension:PORT{[1]|2}:TIME?

Query response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT:PORT1:TIME 1e-3"
20 OUTPUT 717;":SENS1:CORR:EXT:PORT1:TIME?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.EXTension.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.EXTension.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.EXTension.STATe

Description

This command turns ON/OFF or returns the status of the port extension, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the port extension correction
Data type	Boolean type (Boolean)
Range	Select from either of the following. • True or ON: Turns ON the port extension. • False or OFF: Turns OFF the port extension.
Preset value	False or OFF

Examples

```
Dim Ext As Boolean
SCPI.SENSe(1).CORRection.EXTension.STATe = True
Ext = SCPI.SENSe(1).CORRection.EXTension.STATe
```

Related objects

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.DCOFfset

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.LOSS

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.MEASure

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.PORT(Pt)

SCPI.SENSe(Ch).CORRection.EXTension.AUTO.RESet

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).FREQuency(Fq)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).INCLude(Il).STATe

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LDC

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).LOSS(Loss)

SCPI.SENSe(Ch).CORRection.EXTension.PORT(Pt).TIME

Equivalent key

Cal > **Port Extensions** > **Extensions**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:EXTension[:STATe] {ON|OFF|1|0}

:SENSe{[1]-4}:CORRection:EXTension[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:EXT ON"
20 OUTPUT 717;":SENS1:CORR:EXT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.PROPerty

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.PROPerty = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.PROPerty

Description

This command turns ON/OFF or returns the display of the calibration property, for the active trace of selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the display of the calibration property
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the display of the calibration property. • False or OFF: Turns OFF the display of the calibration property.
Preset value	False or OFF

Examples

```
Dim CalProp As Boolean
SCPI.SENSe(1).CORRection.PROPerty = True
CalProp = SCPI.SENSe(1).CORRection.PROPerty
```

Equivalent key

Cal > **Property**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:PROPerty {ON|OFF|1|0}

:SENSe{[1]-4}:CORRection:PROPerty?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:PROP ON"
20 OUTPUT 717;":SENS1:CORR:PROP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.RVELocity.COAX

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.RVELocity.COAX = *Value*

Value = SCPI.SENSe(*Ch*).CORRection.RVELocity.COAX

Description

This command sets/gets the velocity factor for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Velocity factor
Data type	Double precision floating point type (Double)
Range	0.01 to 10
Preset value	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Vel As Double
SCPI.SENSe(1).CORRection.RVELocity.COAX = 0.5
Vel = SCPI.SENSe(1).CORRection.RVELocity.COAX
```

Equivalent key

Cal > **Velocity Factor**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:RVELocity:COAX <numeric>

:SENSe{[1]-4}:CORRection:RVELocity:COAX?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:RVEL:COAX 0.7"
20 OUTPUT 717;":SENS1:CORR:RVEL:COAX?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).CORRection.STATe = *Status*

Status = SCPI.SENSe(*Ch*).CORRection.STATe

Description

This command turns ON/OFF or gets the status of the error correction, for the active trace of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the error correction
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the error correction. • False or OFF: Turns OFF the error correction.
Preset value	False or OFF

Examples

```
Dim Corr As Boolean
SCPI.SENSe(1).CORRection.STATe = True
Corr = SCPI.SENSe(1).CORRection.STATe
```

Equivalent key

Cal > **Correction**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:STATe {ON|OFF|1|0}

:SENSe{[1]-4}:CORRection:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:CORR:STAT ON"
20 OUTPUT 717;":SENS1:CORR:STAT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).CORRection.TYPE(*Tr*)

Object type

Property (**Read Only**)

Syntax

Data = SCPI.SENSe(*Ch*).CORRection.TYPE(*Tr*)

Description

This command reads the information (calibration type, port numbers) of the applied calibration coefficients for the actual error correction, for selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates 5 array data items (the calibration type and the port information to which the calibration is applied). • *Data(0):* The calibration type applied. For detail, refer to the Range section. • *Data(1):* The port number to which the calibration is applied (0 when the calibration type is NONE, GPRO, GPRS, GPRT, or GPS1). • *Data(2)":* The port number to which the calibration is applied (0 when the calibration type is not SOLT2). The array index starts from 0.
Range	One of the following is read out as *Data(0)*. • "ERES": The enhanced response calibration is applied. • "NONE": Nothing is applied. • "RESPO": The response calibration (open) is applied. • "RESPS": The response calibration (short) is applied. • "RESPT": The response calibration (thru) is applied. • "SOLT1": The 1-port calibration is applied. • "SOLT2": The full 2-port calibration is applied. • "GPRO": The gain phase response calibration

	(open) is applied. • "GPRS": The gain phase response calibration (short) is applied. • "GPRT": The gain phase response calibration (thru) is applied. • "GPS1": The gain phase 1-port calibration is applied.
Data type	Variant type (Variant)

Examples

```
Dim CalType As Variant
CalType = SCPI.SENSe(1).CORRection.TYPE(1)
```

Related objects

None.

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:CORRection:TYPE{[1]-4}?

Query response

{ERES|NONE|RESPO|RESPS|RESPT|SOLT1|SOLT2|GPRO|GPRS|GPRT|GPS1},{numeric 1},{numeric 2}<newline><^END>

	Description
ERES	The enhanced response calibration is applied.
NONE	Any calibration is not applied.
RESPO	The response calibration (open) is applied.
RESPS	The response calibration (short) is applied.
RESPT	The response calibration (thru) is applied.

SOLT1	The 1-port calibration is applied.
SOLT2	The 2-port calibration is applied.
GPRO	The gain phase response calibration (open) is applied.
GPRS	The gain phase response calibration (short) is applied.
GPRT	The gain phase response calibration (thru) is applied.
GPS1	The gain phase 1-port calibration is applied.

{numeric 1}:

the calibration port number
(This parameter is 0 when the first parameter is NONE, GPRO, GPRS, GPRT, or GPS1.)

{numeric 2}:

the calibration port number
(This parameter is 0 when the first parameter is not ERES, RESPT, and SOLT2.)

Example of use

```
10 OUTPUT 717;":SENS1:CORR:TYPE1?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).DC.MEASure.CLEar

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).DC.MEASure.CLEar

Description

This command clears the measurement data of DC monitor sweep end, for the selected channel (*Ch*).

Examples

SCPI.SENSe(1).DC.MEASure.CLEar

Related Objects

SCPI.SENSe(Ch).DC.MEASure.ENABle

SCPI.SENSe(Ch).DC.PARameter

Equivalent Key

Meas > **DC Monitor Setup** > **DC Monitor On Sweep End** > **Clear**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:DC:MEASure:CLEar

Example of use

```
10 OUTPUT 717;":SENS1:DC:MEAS:CLE"
```

SCPI.SENSe(*Ch*).DC.MEASure.DATA

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SENSe(*Ch*).DC.MEASure.DATA

Description

This command gets the measurement data of DC monitor sweep end, for the selected channel (*Ch*).

NOTE The unit of *value* is decided by the measurement parameter decided depending on the SCPI.SENSe(Ch).DC.PARameter command.

Variable

Parameter	*Velue*
Description	DC monitor measurement data
Data Type	Double precision floating point type (Double)
Unit	V (voltage) or A (ampere)

Examples

```
Dim DCmon as Double
DCmon = SCPI.SENSe(2).DC.MEASure.DATA
```

Related Objects

SCPI.SENSe(Ch).DC.MEASure.CLEar

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:DC:MEASure:DATA?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:DC:MEAS:DATA?"
20 ENTER 717;A
```

SCPI.SENSe(*Ch*).DC.MEASure.ENABle

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).DC.MEASure.ENABle = *Status*

Status = SCPI.SENSe(*Ch*).DC.MEASure.ENABle

Description

This command turns ON/OFF DC monitor measurement on sweep end, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF the DC monitor measurement on sweep end.
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the DC monitor measurement on sweep end function. • False or OFF: Turns OFF the DC monitor measurement on sweep end function.
Preset Value	False or OFF

Examples

```
Dim StatDCmon as Boolean
SCPI.SENSe(1).DC.MEASure.ENABle = True
StatDCmon = SCPI.SENSe(1).DC.MEASure.ENABle
```

Related Objects

Equivalent Key

Meas > **DC Monitor Setup** > **DC Monitor On Sweep End** > **Monitor**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:DC:MEASure:ENABle {ON|OFF|1|0}

:SENSe{[1]-4}:DC:MEASure:ENABle?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:DC:MEAS:ENAB ON"
20 OUTPUT 717;":SENS1:DC:MEAS:ENAB?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).DC.PARameter

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).DC.PARameter = *Param*

Param = SCPI.SENSe(*Ch*).DC.PARameter

Description

This command sets/gets the port of DC monitor measurement, for the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Port of DC monitor measurement
Data Type	Character string type (String)
Range	Select from either of the following: • "DCV": Sets the DC measurement port to DC output port (V). • "DCI": Sets the DC measurement port to DC output port (A). • "R": Sets the DC measurement port to R-port. • "T": Sets the DC measurement port to T-port.
Preset Value	"DCV"

Examples

```
Dim DCPar as String
SCPI.SENSe(1).DC.PARameter = "DCI"
DCPar = SCPI.SENSe(1).DC.PARameter
```

Related Objects

SCPI.SENSe(Ch).DC.MEASure.DATA

SCPI.SENSe(Ch).DC.MEASure.ENABle

Equivalent Key

Meas > **DC Monitor Setup** > **Function** > **Vdc Bias**|**Idc Bias**|**Vdc R**|**Vdc T**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:DC:PARameter {DCV|DCI|R|T}

:SENSe{[1]-4}:DC:PARameter?

Query Response

{DCV|DCI|R|T} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:DC:PAR DCI"
20 OUTPUT 717;":SENS1:DC:PAR?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).FREQuency.CENTer

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).FREQuency.CENTer = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.CENTer

Description

This command sets/gets the center value of the sweep range of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Center value
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	1.50005G
Unit	Hz (hertz)
Resolution	1E-3 or 5E-4
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Cntr As Double
SCPI.SENSe(1).FREQuency.CENTer = 2E9
Cntr = SCPI.SENSe(1).FREQuency.CENTer
```

Related objects

SCPI.SENSe(Ch).FREQuency.SPAN

Equivalent key

Center

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:FREQuency:CENTer <numeric>

:SENSe{[1]-4}:FREQuency:CENTer?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:FREQ:CENT 2E9"
20 OUTPUT 717;":SENS1:FREQ:CENT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.CW

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).FREQuency.CW = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.CW

Description

This command sets/gets the fixed frequency (CW frequency) for the power/dc bias sweep for the selected channel (*Ch*).

NOTE This object provides the same function as the SCPI.SENSe(Ch).FREQuency.FIXed object.

Variable

Parameter	*Value*
Description	Fixed frequency
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	1E5
Unit	Hz (hertz)
Resolution	1E-3
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim CwFreq As Double
SCPI.SENSe(1).FREQuency.CW = 1E9
CwFreq = SCPI.SENSe(1).FREQuency.CW
```

Related objects

SCPI.SENSe(Ch).FREQuency.FIXed

SCPI.SENSe(Ch).SWEep.TYPE

Equivalent key

Sweep Setup > **Power** > **CW Freq**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:FREQuency[:CW|FIXed] <numeric>

:SENSe{[1]-4}:FREQuency[:CW|FIXed]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:FREQ 1E9"
20 OUTPUT 717;":SENS1:FREQ?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.DATA

Object type

Property (**Read Only**)

Syntax

Data = SCPI.SENSe(Ch).FREQuency.DATA

Description

This command reads the frequencies at all measurement points for the selected channel (Ch).

Variable

Parameter	*Data*
Description	Indicates the array data (frequency) of NOP (number of measurement points). Where n is an integer between 1 and NOP. • *Data(n-1):* Frequency at the n-th measurement point The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim FreqData As Variant
SCPI.SENSe(1).SWEep.POINts = 201
FreqData = SCPI.SENSe(1).FREQuency.DATA
```

Related objects

SCPI.SENSe(Ch).SWEep.POINts

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:FREQuency:DATA?

Query response

{numeric 1},… ,{numeric NOP}<newline><^END>

	Description

{numeric n}	Frequency at the n-th measurement point

Where NOP is the number of measurement points and n is an integer between 1 and NOP.

Example of use

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:FREQ:DATA?"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).FREQuency.FIXed

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).FREQuency.FIXed = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.FIXed

Description

This command sets/gets the fixed frequency (CW frequency) for the power/dc bias sweep for the selected channel (*Ch*).

NOTE This command is similar to SCPI.SENSe(Ch).FREQuency.CW.

Variable

Parameter	*Value*
Description	Fixed frequency
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	1E5
Unit	Hz (hertz)
Resolution	1E-3
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim CwFreq As Double
SCPI.SENSe(1).FREQuency.FIXed = 1E9
CwFreq = SCPI.SENSe(1).FREQuency.FIXed
```

Related objects

SCPI.SENSe(Ch).FREQuency.CW

SCPI.SENSe(Ch).SWEep.TYPE

Equivalent key

Sweep Setup > **Power** > **CW Freq**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:FREQuency[:CW|FIXed] <numeric>

:SENSe{[1]-4}:FREQuency[:CW|FIXed]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:FREQ 1E9"
20 OUTPUT 717;":SENS1:FREQ?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.SPAN

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).FREQuency.SPAN = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.SPAN

Description

This command sets/gets the span value of the sweep range of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Span value
Data type	Double precision floating point type (Double)
Range	0 to 2.999999995E9
Preset value	2.9999E9
Unit	Hz (hertz)
Resolution	1E-3
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Span As Double
SCPI.SENSe(1).FREQuency.SPAN = 1E9
Span = SCPI.SENSe(1).FREQuency.SPAN
```

Related objects

SCPI.SENSe(Ch).FREQuency.CENTer

Equivalent key

Span

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:FREQuency:SPAN <numeric>
:SENSe{[1]-4}:FREQuency:SPAN?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS1:FREQ:SPAN 1E9"
20 OUTPUT 717;":SENS1:FREQ:SPAN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.STARt

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).FREQuency.STARt = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.STARt

Description

This command sets/gets the start value of the sweep range of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Start value
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	1E5
Unit	Hz (hertz)
Resolution	1E-3
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Start As Double
SCPI.SENSe(1).FREQuency.STARt = 100E6
Start = SCPI.SENSe(1).FREQuency.STARt
```

Related objects

SCPI.SENSe(Ch).FREQuency.STOP

Equivalent key

Start

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:FREQuency:STARt <numeric>

:SENSe{[1]-4}:FREQuency:STARt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:FREQ:STAR 100E6"
20 OUTPUT 717;":SENS1:FREQ:STAR?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).FREQuency.STOP

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).FREQuency.STOP = *Value*

Value = SCPI.SENSe(*Ch*).FREQuency.STOP

Description

This command sets/gets the stop value of the sweep range of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Stop value
Data type	Double precision floating point type (Double)
Range	5 to 3E9
Preset value	3E9
Unit	Hz (hertz)
Resolution	1E-3
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Stp As Double
SCPI.SENSe(1).FREQuency.STOP = 3E9
Stp = SCPI.SENSe(1).FREQuency.STOP
```

Related objects

SCPI.SENSe(Ch).FREQuency.STARt

Equivalent key

Stop

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:FREQuency:STOP <numeric>

:SENSe{[1]-4}:FREQuency:STOP?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:FREQ:STOP 100E6"
20 OUTPUT 717;":SENS1:FREQ:STOP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).ROSCillator.SOURce

Object type

Property (**Read Only**)

Syntax

Param = SCPI.SENSe(*Ch*).ROSCillator.SOURce

Description

This command reads whether the external reference signal is inputted to the Ref In connector on the rear panel. The return values are the same for all of Ch.

Variable

Parameter	*Param*
Description	Whether the external reference signal is inputted or not.
Data type	Character string type (String)
Range	Select from either of the following: • "INTernal": The external reference signal is not inputted. • "EXTernal": The external reference signal is inputted.

Examples

```
Dim Ref As String
Ref = SCPI.SENSe(1).ROSCillator.SOURce
```

Equivalent key

Displayed on the instrument status bar (at the bottom of the LCD display).

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:ROSCillator:SOURce?

Query response

{INTernal|EXTernal}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:ROSC:SOUR?"
20 ENTER 717;A$
```

SCPI.SENSe(*Ch*).SEGMent.BWIDth.RESolution.CONTrol

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SEGMent.BWIDth.RESolution.CONTrol = *Status*

Status = SCPI.SENSe(*Ch*).SEGMent.BWIDth.RESolution.CONTrol

Description

This command turns ON/OFF whether to add the IF bandwidth to the segment table, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the IF bandwidth in the segment table
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the IF bandwidth in the segment table. • False or OFF: Turns OFF the IF bandwidth in the segment table.
Preset Value	False or OFF

Examples

```
Dim IFBWStat as Boolean
SCPI.SENSe(1).SEGMent.BWIDth.RESolution.CONTol = True
IFBWStat = SCPI.SENSe(1).SEGMent.BWIDth.RESolution.CONTol
```

Related Objects

Equivalent Key

Sweep Setup > **Edit Segment Table** > **List IFBW**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:SEGMent:BWIDth:RESolution:CONTrol {ON|OFF|1|0}

:SENSe{[1]-4}:SEGMent:BWIDth:RESolution:CONTrol?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SEGM:BWID:RES:CONT ON"
20 OUTPUT 717;":SENS1:SEGM:BWID:RES:CONT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SEGMent.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SEGMent.DATA = *Data*

Data = SCPI.SENSe(*Ch*).SEGMent.DATA

Description

This command creates the segment sweep table of the selected channel (*Ch*).

Variable

Parameter	*Data*
Description	Indicates the array data arranged in the following order (for the segment sweep table); where N is the number of segments (specified with <segm>) and n is an integer between 1 and N. *Data* = {<buf>,<stim>,<ifbw>,<pow>,<del>,<time>,<segm>, <star 1>,<stop 1>,<nop 1>,<ifbw 1>,<pow 1>,<del 1>,<time 1>,... , <star n>,<stop n>,<nop n>,<ifbw n>,<pow n>,<del n>,<time n>,.... , <star N>,<stop N>,<nop N>,<ifbw N>,<pow N>,<del N>,<time N>} Each parameter in the above array data is detailed below: • <buf>: Always specify 5. • <stim>: Stimulus setting mode 0: Specifies with start/stop values 1: Specifies with center/span values • <ifbw>: ON/OFF of the IF bandwidth setting for each segment 0: OFF, 1: ON • <pow>: ON/OFF of the power setting for each segment 0: OFF, 1: ON • <del>: ON/OFF of the sweep delay time setting for each segment

	0: OFF, 1: ON • <time>: ON/OFF of the sweep time setting for each segment 0: OFF, 1: ON • <segm>: Number of segments Specify an integer ranging 1 to 201. • <star n>: Start value/center value of the n-th segment • <stop n>: Stop value/span value of the n-th segment • <nop n>: Number of measurement points of the n-th segment • <ifbw n>: IF bandwidth of the n-th segment is not necessary when the IF bandwidth setting for each segment is OFF (<ifbw>:0). • <pow n>: Power of the n-th segment is not necessary when the power setting for each segment is OFF (<pow>:0). • <del n>: Sweep delay time of the n-th segment is not necessary when the sweep delay time setting for each segment is OFF (<del>:0). • <time n>: Sweep time of the n-th segment is not necessary when the sweep time setting for each segment is OFF (<time>:0).
Data type	Variant type (Variant)
Note	If the necessary amount of array data for the specified number of segments is not available while setting the segment sweep table, an error occurs when its executed and the object is ignored. For <stim>, <ifbw>, <pow>, <del>, and <time>, if the specified value is not the allowable integer, an error occurs when its executed. For <star n>, <stop n>, <nop n>, <ifbw n>, <pow n>, <del n>, and <time n> in the array data, if the specified value is out of the allowable setup range, the minimum value

	(if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim SegmData As Variant
SCPI.SENSe(1).SEGMent.DATA = Array(5,0,0,1,0,0,2, _
100E6,1E9,31,0,2E9,3E9,51,-10)
SegmData = SCPI.SENSe(1).SEGMent.DATA
```

```
Dim SegmData(14) As Variant
Dim Ref As Variant
SegmData(0) = 5
SegmData(1) = 0
SegmData(2) = 0
SegmData(3) = 1
SegmData(4) = 0
SegmData(5) = 0
SegmData(6) = 2
SegmData(7) = 100E6
SegmData(8) = 1E9
SegmData(9) = 31
SegmData(10) = 0
SegmData(11) = 2E9
SegmData(12) = 3E9
SegmData(13) = 51
SegmData(14) = -10
SCPI.SENSe(1).SEGMent.DATA = SegmData
Ref = SCPI.SENSe(1).SEGMent.DATA
```

Related objects

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SENSe(Ch).SEGMent.FMODe

Equivalent key

Sweep Setup > **Edit Segment Table**

Equivalent SCPI command

Syntax

```
:SENSe{[1]-4}:SEGMent:DATA
5,<mode>,<ifbw>,<pow>,<del>,<time>,<segm>,
<star 1>,<stop 1>,<nop 1>,<ifbw 1>,<pow 1>,<del 1>,<time 1>,... ,
<star n>,<stop n>,<nop n>,<ifbw n>,<pow n>,<del n>,<time n>,... ,
<star N>,<stop N>,<nop N>,<ifbw N>,<pow N>,<del N>,<time N>
```

:SENSe{[1]-4}:SEGMent:DATA?

Where N is the number of segments (specified with <segm>) and n is an integer between 1 and N.

Query response

5,{mode},{ifbw},{pow},{del},{time},{segm},
{star 1},{stop 1},{nop 1},{pow 1},{del 1},{time 1},... ,
{star n},{stop n},{nop n},{pow n},{del n},{time n},... ,
{star N},{stop N},{nop N},{pow N},{del N},{time N}<newline><^END>

Example of use

```
10 DIM H(1:3,1:4)
20 OUTPUT 717;":SENS1:SEGM:DATA 5,0,1,0,0,0,3,";
30 OUTPUT 717;"1E9,3E9,11,70e3,";
40 OUTPUT 717;"3E9,4E9,51,7e3,";
50 OUTPUT 717;"4E9,6E9,11,70e3"
60 OUTPUT 717;":SENS1:SEGM:DATA?"
70 ENTER 717;A,B,C,D,E,F,G,H(*)
```

SCPI.SENSe(*Ch*).SEGMent.FMODe

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SEGMent.FMODe = *Param*

Param = SCPI.SENSe(*Ch*).SEGMent.FMODe

Description

This command sets/gets the stimulus setting mode of stimulus range setting in segment table, for the selected channel (*Ch*). This command is the same operation as the second parameter of SCPI.SENSe(Ch).SEGMent.DATA command.

Variable

Parameter	*Param*
Description	Stimulus setting mode in segment table
Data Type	Character string type (String)
Range	Select from either of the following: • "ENDS": Specified stimulus setting mode to "Start/Stop". • "SPAN": Specified stimulus setting mode to "Center/Span".
Preset Value	"ENDS"

Examples

```
Dim FreqMode as String
SCPI.SENSe(1).SEGMent.FMODe = "SPAN"
FreqMode = SCPI.SENSe(1).SEGMent.FMODe
```

Related Objects

SCPI.SENSe(Ch).SEGMent.DATA

Equivalent Key

Sweep Setup > **Edit Segment Table** > **Freq Mode** > **Start/Stop|Center/Span**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:SEGMent:FMODe {ENDS|SPAN}

:SENSe{[1]-4}:SEGMent:FMODe?

Query Response

{ENDS|SPAN} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SEGM:FMOD SPAN"
20  OUTPUT 717;":SENS1:SEGM:FMOD?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).SEGMent.POWer.LEVel.CONTrol

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SEGMent.POWer.LEVel.CONTrol = *Status*

Status = SCPI.SENSe(*Ch*).SEGMent.POWer.LEVel.CONTrol

Description

This command turns ON/OFF whether to add the power level to the segment table, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the power level in the segment table
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the power level in the segment table. • False or OFF: Turns OFF the power level in the segment table.
Preset Value	False or OFF

Examples

```
Dim PowStat as Boolean
SCPI.SENSe(1).SEGMent.POWer.LEVel.CONTol = True
PowStat = SCPI.SENSe(1).SEGMent.POWer.LEVel.CONTol
```

Related Objects
Equivalent Key

Sweep Setup > **Edit Segment Table** > **List Power**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:SEGMent:POWer:LEVel:CONTrol {ON|OFF|1|0}

:SENSe{[1]-4}:SEGMent:POWer:LEVel:CONTrol?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SEGM:POW:LEV:CONT ON"
20 OUTPUT 717;":SENS1:SEGM:POW:LEV:CONT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SEGMent.SWEep.DELay.CONTrol

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SEGMent.SWEep.DELay.CONTrol = *Status*

Status = SCPI.SENSe(*Ch*).SEGMent.SWEep.DELay.CONTrol

Description

This command turns ON/OFF whether to add the sweep delay time to the segment table, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the sweep delay time in the segment table
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the sweep delay time in the segment table. • False or OFF: Turns OFF the sweep delay time in the segment table.
Preset Value	False or OFF

Examples

```
Dim DelStat as Boolean
SCPI.SENSe(1).SEGMent.SWEep.DELay.CONTol = True
DelStat = SCPI.SENSe(1).SEGMent.SWEep.DELay.CONTol
```

Related Objects
Equivalent Key

Sweep Setup > **Edit Segment Table** > **List Delay**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:SEGMent:SWEep:DELay:CONTrol {ON|OFF|1|0}

:SENSe{[1]-4}:SEGMent:SWEep:DELay:CONTrol?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SEGM:SWE:DEL:CONT ON"
20 OUTPUT 717;":SENS1:SEGM:SWE:DEL:CONT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SEGMent.SWEep.POINts

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SENSe(*Ch*).SEGMent.SWEep.POINts

Description

This command reads the total number of the measurement points of all segments, for the segment sweep table of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Total number of measurement points of all the segments
Data type	Long integer type (Long)
Preset Value	2

Examples

```
Dim SegmPoin As Long
SegmPoin = SCPI.SENSe(1).SEGMent.SWEep.POINts
```

Related objects

SCPI.SENSe(Ch).SEGMent.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:SEGMent:SWEep:POINts?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SEGM:SWE:POIN?"
20 ENTER 717;A
```

SCPI.SENSe(Ch).SEGMent.SWEep.TIME.CONTrol

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SEGMent.SWEep.TIME.CONTrol = *Status*

Status = SCPI.SENSe(*Ch*).SEGMent.SWEep.TIME.CONTrol

Description

This command turns ON/OFF whether to add the sweep time to the segment table, for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the sweep time in the segment table
Data Type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the sweep time in the segment table. • False or OFF: Turns OFF the sweep time in the segment table.
Preset Value	False or OFF

Examples

```
Dim SwTimeStat as Boolean
SCPI.SENSe(1).SEGMent.SWEep.TIME.CONTol = True
SwTimeStat = SCPI.SENSe(1).SEGMent.SWEep.TIME.CONTol
```

Related Objects
Equivalent Key

Sweep Setup > **Edit Segment Table** > **List Time**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:SEGMent:SWEep:TIME:CONTrol {ON|OFF|1|0}

:SENSe{[1]-4}:SEGMent:SWEep:TIME:CONTrol?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SEGM:SWE:TIME:CONT ON"
20 OUTPUT 717;":SENS1:SEGM:SWE:TIME:CONT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SEGMent.SWEep.TIME.DATA

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SENSe(*Ch*).SEGMent.SWEep.TIME.DATA

Description

This command reads the total sweep time (including sweep delay time) of all the segments, for the segment sweep table of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Total sweep time of all segments
Data type	Double precision floating point type (Double)

Examples

```
Dim SegmTime As Double
SegmTime = SCPI.SENSe(1).SEGMent.SWEep.TIME.DATA
```

Related objects

SCPI.SENSe(Ch).SEGMent.DATA

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:SEGMent:SWEep:TIME[:DATA]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SEGM:SWE:TIME?"
20 ENTER 717;A
```

SCPI.SENSe(Ch).SWEep.BAND.WAIT

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(Ch).SWEep.BAND.WAIT = *Value*

Value = SCPI.SENSe(Ch).SWEep.BAND.WAIT

Description

This command sets/gets the additional band wait time for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Waiting time
Data Type	Double precision floating point type (Double)
Range	0 to 1
Preset Value	0
Unit	sec
Resolution	100u

Examples

```
Dim Var as Double
Var= 0.1
SCPI.SENSe(1).SWEep.BAND.WAIT = Var
Var = SCPI.SENSe(1).SWEep.BAND.WAIT
```

Equivalent Key

System > **Service Menu** > **Band Wait**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:SWEep:BAND:WAIT <numeric>

:SENSe{[1]-4}:SWEep:BAND:WAIT?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SWE:BAND:WAIT 0.1"
20 OUTPUT 717;":SENS1:SWE:BAND:WAIT ?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.DELay

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SWEep.DELay = *Value*

Value = SCPI.SENSe(*Ch*).SWEep.DELay

Description

This command sets/gets the sweep delay time of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Sweep delay time
Data type	Double precision floating point type (Double)
Range	0 to 1
Preset value	0
Unit	s (second)
Resolution	0.001
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim SweDel As Double
SCPI.SENSe(1).SWEep.DELay = 0.05
SweDel = SCPI.SENSe(1).SWEep.DELay
```

Equivalent key

Sweep Setup > **Sweep Delay**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:SWEep:DELay <numeric>

:SENSe{[1]-4}:SWEep:DELay?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SWE:DEL 0.05"
20 OUTPUT 717;":SENS1:SWE:DEL?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.POINts

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SWEep.POINts = *Value*

Value = SCPI.SENSe(*Ch*).SWEep.POINts

Description

This command sets/gets the number of measurement points of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of measurement points
Data type	Long integer type (Long)
Range	2 to 1601
Preset value	201
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Nop As Long
SCPI.SENSe(1).SWEep.POINts = 801
Nop = SCPI.SENSe(1).SWEep.POINts
```

Equivalent key

Sweep Setup > **Points**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:SWEep:POINts <numeric>

:SENSe{[1]-4}:SWEep:POINts?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SWE:POIN 801"
20 OUTPUT 717;":SENS1:SWE:POIN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.TIME.AUTO

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SWEep.TIME.AUTO = *Status*

Status = SCPI.SENSe(*Ch*).SWEep.TIME.AUTO

Description

This command sets/gets whether to automatically set the sweep time of the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the auto setting of the sweep time
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the auto setting. • False or OFF: Turns OFF the auto setting.
Preset value	True or ON

Examples

```
Dim SweAuto As Boolean
SCPI.SENSe(1).SWEep.TIME.AUTO = False
SweAuto = SCPI.SENSe(1).SWEep.TIME.AUTO
```

Related objects

SCPI.SENSe(Ch).SWEep.TIME.DATA

Equivalent key

Sweep Setup > **Sweep Time Auto**

NOTE When performing the operation from the front panel, the auto setting of the sweep time is turned ON by setting the sweep time to 0 s.

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:SWEep:TIME:AUTO {ON|OFF|1|0}

:SENSe{[1]-4}:SWEep:TIME:AUTO?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SWE:TIME:AUTO ON"
20 OUTPUT 717;":SENS1:SWE:TIME:AUTO?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.TIME.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SWEep.TIME.DATA = *Value*

Value = SCPI.SENSe(*Ch*).SWEep.TIME.DATA

Description

This command sets/gets the sweep time of the selected channel (*Ch*).

NOTE Before using this object to set the sweep time, turns OFF the auto setting of the sweep time (specify False with the SCPI.SENSe(Ch).SWEep.TIME.AUTO object).

Variable

Parameter	*Value*
Description	Sweep time
Data type	Double precision floating point type (Double)
Range	Varies depending on the measurement conditions
Preset value	Varies depending on the measurement conditions
Unit	s (second)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim SweTime As Double
SCPI.SENSe(1).SWEep.TIME.AUTO = False
SCPI.SENSe(1).SWEep.TIME.DATA = 1.5
SweTime = SCPI.SENSe(1).SWEep.TIME.DATA
```

Related objects

SCPI.SENSe(Ch).SWEep.TIME.AUTO

Equivalent key

Sweep Setup > Sweep Time

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:SWEep:TIME[:DATA] <numeric>

:SENSe{[1]-4}:SWEep:TIME[:DATA]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SWE:TIME 1.5"
20 OUTPUT 717;":SENS1:SWE:TIME?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).SWEep.TYPE

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).SWEep.TYPE = *Param*

Param = SCPI.SENSe(*Ch*).SWEep.TYPE

Description

This command sets/gets the sweep type of the selected channel (*Ch*).

Variable

Parameter	*Param*
Description	Sweep type
Data type	Character string type (String)
Range	Select from either of the following: • "LINear": Sets the sweep type to the linear sweep. • "LOGarithmic": Sets the sweep type to the log sweep. • "SEGMent": Sets the sweep type to the segment sweep. • "POWer": Sets the sweep type to the power sweep. • "BIAS": Sets the sweep type to the DC bias sweep.
Preset value	"LINear"

Examples

```
Dim SweType As String
SCPI.SENSe(1).SWEep.TYPE = "segm"
SweType = SCPI.SENSe(1).SWEep.TYPE
```

Equivalent key

Sweep Setup > **Sweep Type** > **Lin Freq**|**Log Freq**|**Segment**|**Power Sweep**|**DC Bias Sweep**

Equivalent SCPI command

Syntax

:SENSe{[1]-4}:SWEep:TYPE {LINear|LOGarithmic|SEGMent|POWer|BIAS}

:SENSe{[1]-4}:SWEep:TYPE?

Query response

{LIN|LOG|SEGM|POW|BIAS}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:SWE:TYPE SEGM"
20 OUTPUT 717;":SENS1:SWE:TYPE?"
30 ENTER 717;A$
```

SCPI.SENSe.CORRection.IMPedance.INPut.MAGNitude

Object type

Property (**Read-Write**)

Syntax

SCPI.SENSe.CORRection.IMPedance.INPut.MAGNitude = *Value*

Value = SCPI.SENSe.CORRection.IMPedance.INPut.MAGNitude

Description

This command sets/gets the system characteristic impedance (Z0) value.

NOTE The system impedance is a common parameter in all the channels.

Variable

Parameter	*Value*
Description	System Z0 value
Data type	Double precision floating point type (Double)
Range	1E-3 to 1000
Preset value	50
Unit	Ω
Resolution	0.001
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim SysZ0 As Double
SCPI.SENSe.CORRection.IMPedance.INPut.MAGNitude = 75
SysZ0 = SCPI.SENSe.CORRection.IMPedance.INPut.MAGNitude
```

Equivalent key

Cal > **Set Z0**

Equivalent SCPI command

Syntax

```
:SENSe:CORRection:IMPedance[:INPut][:MAGNitude] <numeric>
:SENSe:CORRection:IMPedance[:INPut][:MAGNitude]?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SENS:CORR:IMP 75"
20 OUTPUT 717;":SENS:CORR:IMP?"
30 ENTER 717;A
```

SCPI.SENSe(Ch).Z.COMPensation.CLEar

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(Ch).Z.COMPensation.CLEar

Description

This command clears the fixture compensation table for the selected channel.

Examples

SCPI.SENSe(1).Z.COMPensation.CLEar

Related Objects

SCPI.SENSe(Ch).Z.CORRection.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Clear** > **OK**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:CLEar

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:CLE"
```

SCPI.SENSe(*Ch*).Z.COMPensation.COEFficient.DATA

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COEFficient.DATA(*Std*) = *Array*

Array = SCPI.SENSe(*Ch*).Z.COMPensation.COEFficient.DATA(*Std*)

Description

This command sets/gets the measured impedance data for each fixture compensation standard for the selected channel (*Ch*).

Variable

Parameter	*Std*
Description	Standrad
Data Type	Character string type (String)
Range	OPEN: Open SHORt Short LOAD: Load

Parameter	*Array*
Description	Indicates the array data (corrected data array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. *Data(n×2-2)* Real part of data (complex number) at the n-th measurement point. *Data(n×2-1)* Imaginary part of data (complex number) at the n-th measurement point. The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim CoefAray As Variant, NoOfPoint  As Long, dmy As Long

MsgBox "Connect Open"
SCPI.SENSe(1).Z.COMPensation.COLLect.ACQuire = "OPEN"
dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).Z.COMPensation.COLLect.SAVE
SCPI.SENSe(2).Z.COMPensation.COLLect.ACQuire = "OPEN"
dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(2).Z.COMPensation.COLLect.SAVE

NoOfPoint = SCPI.SENSe.Z.COMPensation.COEFficient.POINts("OPEN")

ReDim CoefAray(NoOfPoint)
CoefAray = SCPI.SENSe(1).Z.COMPensation.COEFficient.DATA("OPEN")
SCPI.SENSe(2).Z.COMPensation.COEFficient.DATA("OPEN") = CoefAray
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

SCPI.SENSe(Ch).Z.COMPensation.COEFficient.POINts

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COEFficient[:DATA] {OPEN|SHORt|LOAD}, <numeric 1>, ... ,<numeric NOP×2>

:SENSe{[1]-4}:Z:COMPensation:COEFficient[:DATA]? {OPEN|SHORt|LOAD}

Query Response

<numeric 1>, ... ,<numeric NOP×2><newline><^END>

	Description
<numeric n×2-1>	Real part of data (complex number) at the n-th measurement point.
<numeric n×2>	Imaginary part of data (complex number) at the n-th measurement point.

Because the calibration coefficient array is expressed by a complex number, the real part and the imaginary part of one measurement point are returned and obtained as a value. Here, NOP is the number of measurement points and n is an integer between 1 and NOP.

Example of use

```
10 OUTPUT 717;":SENS:Z:COMP:COEF OPEN, 50,0, 50.1, 0, 50.2, 0"
20 OUTPUT 717;":SENS:Z:COMP:COEF? OPEN"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).Z.COMPensation.COEFficient.POINts

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COEFficient.POINts(*Std*)

Description

This command returns the size of array of fixture compensation coefficient data of Impedance measurement which is retuned by SCPI.SENSe(Ch).Z.COMPensation.COEFficient.DATA.

Variable

Parameter	*Std*
Description	Standrad
Data Type	Character string type (String)
Range	OPEN: Open SHORt Short LOAD: Load

Parameter	*Value*
Description	Size of array
Data Type	Long integer type (Long)
Range	0 to 1601
Preset Value	0
Unit	- -
Resolution	- -

Examples

See SCPI.SENSe(Ch).Z.COMPensation.COEFficient.DATA

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COEFficient.DATA

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COEFficient:POINts? {OPEN|SHORt|LOAD}

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COEF:POIN? OPEN"
20 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.ACQuire

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.ACQuire = *Standard*

Description

This command executes the open, short or load of the fixture compensation for the selected channel (*Ch*).

Variable

Parameter	*Standard*
Description	Compensation Type
Data Type	Character string type (String)
Range	OPEN: Open SHORt: Short LOAD: Load
Preset Value	-
Unit	-
Resolution	-

Examples

```
Dim Dmy as Long
SCPI.SENSe(1).Z.COMPensation.COLLect.ACQuire = "OPEN"
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).Z.COMPensation.COLLect.SAVE
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.SAVE

SCPI.SENSe(Ch).Z.COMPensation.CLEar

Equivalent Key

Calibration > **Fixture Compen** > **Compensate** > **Open**

Calibration > **Fixture Compen** > **Compensate** > **Short**

Calibration > **Fixture Compen** > **Compensate** > **Load**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect[:ACQuire] {OPEN|SHORt|LOAD}

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL OPEN"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
40 OUTPUT 717;":SENS1:Z:COMP:COLL:SAVE"
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.SAVE

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.SAVE

Description

This command calculates the calibration coefficients for fixture compensation for the selected channel (*Ch*).

Examples

See SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compensate** > **Done**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:SAVE

Example of use

See :SENSe{[1]-4}:Z:COMPensation:COLLect[:ACQuire]

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.CP

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.CP = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.CP

Description

This command sets/gets the Cp value of Load Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Cp value of Load Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	F
Resolution	-

Examples

```
Dim Var as Double
Var= 1E-13
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.LOAD.CP = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.LOAD.CP
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.LOAD.LS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.LOAD.RS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

SCPI.SENSe(Ch).Z.COMPensation.COLLect.SAVE

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Load Cp**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:LOAD:CP <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:LOAD:CP?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:LOAD:CP 1E-13"
20 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:LOAD:CP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.LS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.LS = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.LS

Description

This command sets/gets the Ls value of Load Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Ls value of Load Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	H
Resolution	-

Examples

```
Dim Var as Double
Var= 1E-10
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.LOAD.LS = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.LOAD.LS
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.LOAD.CP

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.LOAD.RS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Load Ls**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:LOAD:LS <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:LOAD:LS?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENSe1:Z:COMPensation:COLLect:STAN:LOAD:LS 1E-10"
20 OUTPUT 717;":SENSe1:Z:COMPensation:COLLect:STAN:LOAD:LS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.RS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.RS = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.LOAD.RS

Description

This command sets/gets the Rs value of Load Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Rs value of Load Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	50
Unit	Ω
Resolution	-

Examples

```
Dim Var as Double
Var= 50.5
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.LOAD.RS = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.LOAD.RS
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.LOAD.CP

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.LOAD.RS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Load Rs**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:LOAD:RS <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:LOAD:RS?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENSe1:Z:COMP:COLL:STAN:LOAD:RS 50.5"
20 OUTPUT 717;":SENSe1:Z:COMP:COLL:STAN:LOAD:RS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.CP

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.CP = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.CP

Description

This command sets/gets the Cp value of Open Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Cp value of Open Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	F
Resolution	-

Examples

```
Dim Var as Double
Var= 1E-13
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.OPEN.CP = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.OPEN.CP
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.OPEN.GS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.OPEN.LS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Open Cp**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:OPEN:CP <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:OPEN:CP?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:OPEN:CP 1E-13"
20 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:OPEN:CP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.LS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.LS = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.LS

Description

This command sets/gets the Ls value of Open Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Ls value of Open Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	H
Resolution	-

Examples

```
Dim Var as Double
Var= 1E-10
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.OPEN.LS = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.OPEN.LS
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.OPEN.GS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.OPEN.CP

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Open Ls**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:OPEN:LS <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:OPEN:LS?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:OPEN:LS 1E-10"
20 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:OPEN:LS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.GS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.GS = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.OPEN.GS

Description

This command sets/gets the Gs value of Open Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Gs value of Open Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	S
Resolution	-

Examples

```
Dim Var as Double
Var= 0.01
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.OPEN.GS = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.OPEN.GS
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.OPEN.CP

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.OPEN.LS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Open Gs**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:OPEN:GS <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:OPEN:GS?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:OPEN:GS 0.01"
20 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:OPEN:GS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.CP

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.CP = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.CP

Description

This command sets/gets the Cp value of Short Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Cp value of Short Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	F
Resolution	-

Examples

```
Dim Var as Double
Var= 1E-13
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.SHORt.CP = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.SHORt.CP
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.SHORt.LS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.SHORt.RS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Short Cp**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:SHORt:CP <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:SHORt:CP?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:SHORt:CP 1E-13"
20 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:SHORt:CP?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.LS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.LS = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.LS

Description

This command sets/gets the Ls value of Short Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Ls value of Short Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	H
Resolution	-

Examples

```
Dim Var as Double
Var= 1E-10
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.SHORt.LS = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.SHORt.LS
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.SHORt.CP

SCPI.SENSe(Ch).Z.COMPensation.COLLect.STAN.SHORt.RS

SCPI.SENSe(Ch).Z.COMPensation.COLLect.ACQuire

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Short Ls**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:SHORt:LS <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:SHORt:LS?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:SHORt:LS 1E-10"
20 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:SHORt:LS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.RS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.RS = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.COLLect.STAN.SHORt.RS

Description

This command sets/gets the Rs value of Short Standard for the fixture compensation for the selected channel (*Ch*). This is independent from the calibration kit.

Variable

Parameter	*Value*
Description	Rs value of Short Standard
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	Ω
Resolution	-

Examples

```
Dim Var as Double
Var= 0.001
SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.SHORt.RS = Var
Var = SCPI.SENSe(1).Z.COMPensation.COLLect.STAN.SHORt.RS
```

Related Objects

Equivalent Key

Calibration > **Fixture Compen** > **Compen STDs** > **Short Rs**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:SHORt:RS <numeric>

:SENSe{[1]-4}:Z:COMPensation:COLLect:STAN:SHORt:RS?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:SHORt:RS 0.001"
20 OUTPUT 717;":SENS1:Z:COMP:COLL:STAN:SHORt:RS?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.EDELay.TIME

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.EDELay.TIME = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.EDELay.TIME

Description

This command sets/gets the Z port extension delay value for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Z port extension
Data Type	Double precision floating point type (Double)
Range	-1M ~ 1M
Preset Value	0
Unit	sec
Resolution	-

Examples

```
Dim Var as Double
Var= 0.001
SCPI.SENSe(1).Z.COMPensation.EDELay.TIME = Var
Var = SCPI.SENSe(1).Z.COMPensation.EDELay.TIME
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.FIXTure.EDELay.USER.DISTance

Equivalent Key

Calibration > **Fixture Compen** > **Z Port Extension**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:EDELay:TIME <numeric>

:SENSe{[1]-4}:Z:COMPensation:EDELay:TIME?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENSe1:Z:COMP:EDEL:TIME 0.001"
20 OUTPUT 717;":SENSe1:Z:COMP:EDEL:TIME?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.EDELay.MODel.DISTance

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.EDELay.MODel.DISTance("NONE"|"FXT16191A"|"FXT16192A"|"FXT16193A"|"FXT16194A"|"FXT16196A"|"FXT16196B"|"FXT16196C"|"FXT16197")

Description

This command gets the registered electrical length of specified fixture for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Electrical Length
Data Type	Double precision floating point type (Double)
Range	-
Preset Value	-
Unit	meter
Resolution	-

Examples

```
Dim Var as Double
Var = SCPI.SENSe(1).Z.COMPensation.FIXTure.EDELay.MODel.DISTance ("FXT16191A")
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.FIXTure.SELect

Equivalent Key

Calibration > **Fixture Compen** > **Fixture**

This command returns the electrical length value on the softkey for each fixture.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:FIXTure:EDELay:MODel:DISTance {NONE|FXT16191A|FXT16192A|FXT16193A|FXT16194A|FXT16196A|FXT16196B|FXT16196C|FXT16197A}

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:FIXT:EDEL:MOD:DIST? FXT16191A"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.EDELay.USER.DISTance

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.EDELay.USER.DISTance = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.EDELay.USER.DISTance

Description

This command sets/gets the electrical lenght of user fixture for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Electrical Length
Data Type	Double precision floating point type (Double)
Range	-1k ~ 1k
Preset Value	0
Unit	m
Resolution	-

Examples

```
Dim Var as Double
Var= 0.015
SCPI.SENSe(1).Z.COMPensation.FIXTure.EDELay.USER.DISTance = Var
Var = SCPI.SENSe(1).Z.COMPensation.FIXTure.EDELay.USER.DISTance
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.FIXTure.SELect

Equivalent Key

Calibration > **Fixture Compen** > **Modify User Fixt.** > **User Fixture**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:FIXTure:EDELay:USER:DISTance <numeric>

:SENSe{[1]-4}:Z:COMPensation:FIXTure:EDELay:USER:DISTance?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENSe1:Z:COMPensation:FIXTure:EDELay:USER:DISTance 0.015"
20 OUTPUT 717;":SENSe1:Z:COMPensation:FIXTure:EDELay:USER:DISTance?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.SELect

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.SELect = *Fixture*

Fixture = SCPI.SENSe(*Ch*).Z.COMPensation.FIXTure.SELect

Description

This command sets/gets the fixture model number to compensate the electrical length for the selected channel (*Ch*). Whne user fixture is selected, the electrical length for user fixture should be specified by SCPI.SENSe(Ch).Z.COMPensation.FIXTure.EDELay.USER.DISTance.

Variable

Parameter	*Fixture*
Description	Fixture model number
Data Type	Character string type (String)
Range	NONE|FXT16191A|FXT16192A|FXT16193A|FXT16194A|FXT16196A|FXT16196B|FXT16196C|FXT16197A|USER
Preset Value	NONE
Unit	-
Resolution	-

Examples

```
Dim Fixture as String
Fixture= "FXT16191A"
SCPI.SENSe(1).Z.COMPensation.FIXTure.SELect = Fixture
Fixture = SCPI.SENSe(1).Z.COMPensation.FIXTure.SELect
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.FIXTure.EDELay.USER.DISTance

SCPI.SENSe(Ch).Z.COMPensation.FIXTure.EDELay.MODel.DISTance

Equivalent Key

Calibration > **Fixture Compen** > **Fixture**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:FIXTure[:SELect] {NONE|FXT16191A|FXT16192A|FXT16193A|FXT16194A|FXT16196A|FXT16196B|FXT16196C|FXT16197A|USER}

:SENSe{[1]-4}:Z:COMPensation:FIXTure[:SELect]?

Query Response

{NONE|FXT16191A|FXT16192A|FXT16193A|FXT16194A|FXT16196A|FXT16196B|FXT16196C|FXT16197A|USER} <newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:FIXT FXT16191A"
20 OUTPUT 717;":SENS1:Z:COMP:FIXT?"
30 ENTER 717;A$
```

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.ALL

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.ALL = *Status*

Status = SCPI.SENSe(*Ch*).Z.COMPensation.STATe.ALL

Description

This command turns On/Off the fixture compensation for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Fixture compensation on/off
Data Type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON(1): Turns ON. • False or OFF(0): Turns OFF.
Preset Value	OFF
Unit	-
Resolution	-

Examples

```
Dim CompStatus as Boolean
SCPI.SENSe(1).Z.COMPensation.STATe.ALL = True
CompStatus = SCPI.SENSe(1).Z.COMPensation.STATe.ALL
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.STATe.LOAD

SCPI.SENSe(Ch).Z.COMPensation.STATe.OPEN

SCPI.SENSe(Ch).Z.COMPensation.STATe.SHORt

Equivalent Key

Calibration > **Fixture Compen** > **Fixture Compen**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:STATe[:ALL] {ON|OFF|1|0}

:SENSe{[1]-4}:Z:COMPensation:STATe[:ALL]?

Query Response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:STAT ON"
20 OUTPUT 717;":SENS1:Z:COMP:STAT?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.LOAD

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.LOAD = *Status*

Status = SCPI.SENSe(*Ch*).Z.COMPensation.STATe.LOAD

Description

This command turns On/Off the load of fixture compensation for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Load compensation on/off
Data Type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON(1): Turns ON. • False or OFF(0): Turns OFF.
Preset Value	OFF
Unit	-
Resolution	-

Examples

```
Dim LoadCompStatus as Boolean
SCPI.SENSe(1).Z.COMPensation.STATe.LOAD = True
LoadCompStatus = SCPI.SENSe(1).Z.COMPensation.STATe.LOAD
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.STATe.ALL

SCPI.SENSe(Ch).Z.COMPensation.STATe.OPEN

SCPI.SENSe(Ch).Z.COMPensation.STATe.SHORt

Equivalent Key

Calibration > **Fixture Compen** > **Compensate** > **Compen Load**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:STATe:LOAD {ON|OFF|1|0}

:SENSe{[1]-4}:Z:COMPensation:STATe:LOAD?

Query Response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENSe1:Z:COMPensation:STATe:LOAD ON"
20 OUTPUT 717;":SENSe1:Z:COMPensation:STATe:LOAD?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.OPEN

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.OPEN = *Status*

Status = SCPI.SENSe(*Ch*).Z.COMPensation.STATe.OPEN

Description

This command turns On/Off the open of fixture compensation for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Open compensation on/off
Data Type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON(1): Turns ON. • False or OFF(0): Turns OFF.
Preset Value	OFF
Unit	-
Resolution	-

Examples

```
Dim OpenCompStatus as Boolean
SCPI.SENSe(1).Z.COMPensation.STATe.OPEN = Ture
OpenCompStatus = SCPI.SENSe(1).Z.COMPensation.STATe.OPEN
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.STATe.ALL

SCPI.SENSe(Ch).Z.COMPensation.STATe.LOAD

SCPI.SENSe(Ch).Z.COMPensation.STATe.SHORt

Equivalent Key

Calibration > **Fixture Compen** > **Compensate** > **Compen Open**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:STATe:OPEN {ON|OFF|1|0}

:SENSe{[1]-4}:Z:COMPensation:STATe:OPEN?

Query Response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:STAT:OPEN ON"
20 OUTPUT 717;":SENS1:Z:COMP:STAT:OPEN?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.SHORt

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.COMPensation.STATe.SHORt = *Value*

Value = SCPI.SENSe(*Ch*).Z.COMPensation.STATe.SHORt

Description

This command turns On/Off the short of fixture compensation for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	Short compensation on/off
Data Type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON(1): Turns ON. • False or OFF(0): Turns OFF.
Preset Value	OFF
Unit	-
Resolution	-

Examples

```
Dim ShortCompStatus as Boolean
SCPI.SENSe(1).Z.COMPensation.STATe.SHORt = Ture
ShortCompStatus= SCPI.SENSe(1).Z.COMPensation.STATe.SHORt
```

Related Objects

SCPI.SENSe(Ch).Z.COMPensation.STATe.ALL

SCPI.SENSe(Ch).Z.COMPensation.STATe.LOAD

SCPI.SENSe(Ch).Z.COMPensation.STATe.OPEN

Equivalent Key

Calibration > **Fixture Compen** > **Compensate** > **Compen Short**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:COMPensation:STATe:SHORt {ON|OFF|1|0}

:SENSe{[1]-4}:Z:COMPensation:STATe:SHORt?

Query Response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:COMP:STAT:SHOR ON"
20 OUTPUT 717;":SENS1:Z:COMP:STAT:SHORt?"
30 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.CORRection.COEFficient.DATA

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.CORRection.COEFficient.DATA(*Std*) = *Array*

Array = SCPI.SENSe(*Ch*).Z.CORRection.COEFficient.DATA(*Std*)

Description

This command sets/gets the calibration coefficient data of impedance calibration for the selected channel. Before writing the calibration coefficient array, the calibration should be performed in advance in that channel with the same setting. After writing the calibration coefficient array, the written value becomes effective immediately.

Variable

Parameter	*Std*
Description	Standrad
Data Type	String
Range	OPEN: Open SHORt Short LOAD: Load LOAD2: Low-loss capacitor

Parameter	*Array*
Description	Indicates the array data (corrected data array) of NOP (number of measurement points)×2. Where n is an integer between 1 and NOP. The size of array, NOP, is the number of points when the calibration is executed. *Data(n×2-2)* Real part of data (complex number) at the n-th measurement point. *Data(n×2-1)* Imaginary part of data (complex number) at the n-th measurement point.

	The index of the array starts from 0.
Data type	Variant type (Variant)

Examples

```
Dim CoefArray as Variant, NoOfPoint as Long
Dim Dmy As Long

MsgBox "Connect Open"
SCPI.SENSe(1).Z.CORRection.COLLect.ACQuire = "OPEN"
Dmy = SCPI.IEEE4882.OPC
MsgBox "Connect Short"
SCPI.SENSe(1).Z.CORRection.COLLect.ACQuire = "SHORT"
Dmy = SCPI.IEEE4882.OPC
MsgBox "Connect Load"
SCPI.SENSe(1).Z.CORRection.COLLect.ACQuire = "LOAD"
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(1).Z.CORRection.COLLect.SAVE

MsgBox "Connect Open"
SCPI.SENSe(2).Z.CORRection.COLLect.ACQuire = "OPEN"
Dmy = SCPI.IEEE4882.OPC
MsgBox "Connect Short"
SCPI.SENSe(2).Z.CORRection.COLLect.ACQuire = "SHORT"
Dmy = SCPI.IEEE4882.OPC
MsgBox "Connect Load"
SCPI.SENSe(2).Z.CORRection.COLLect.ACQuire = "LOAD"
Dmy = SCPI.IEEE4882.OPC
SCPI.SENSe(2).Z.CORRection.COLLect.SAVE

NoOfPoint = SCPI.SENSe(1).Z.CORRection.COEFficient.POINts
Redim CoefArry(NoOfPoint)
CoefArray = SCPI.SENSe(2).Z.CORRection.COEFficient.DATA("OPEN")
SCPI.SENSe(1).Z.CORRection.COEFficient.DATA("OPEN") = CoefArray
```

Related Objects

SCPI.SENSe(Ch).Z.CORRection.COLLect.ACQuire

SCPI.SENSe(Ch).Z.CORRection.COEFficient.POINts

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:CORRection:COEFficient[:DATA]
{OPEN|SHORt|LOAD|LOAD2},<numeric 1>, ... ,<numeric NOP×2>

:SENSe{[1]-4}:Z:CORRection:COEFficient[:DATA]?

Query Response

<numeric 1>, ... ,<numeric NOP×2><newline><^END>

	Description
<numeric n×2-1>	Real part of data (complex number) at the n-th measurement point.
<numeric n×2>	Imaginary part of data (complex number) at the n-th measurement point.

Because the calibration coefficient array is expressed by a complex number, the real part and the imaginary part of one measurement point are returned and obtained as a value. Here, NOP is the number of measurement points and n is an integer between 1 and NOP.

Example of use

```
10 DIM A(1:201)
20 OUTPUT 717;":SENS1:Z:CORR:COEF? OPEN"
30 ENTER 717;A(*)
```

SCPI.SENSe(*Ch*).Z.CORRection.COEFficient.POINts

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SENSe(*Ch*).Z.CORRection.COEFficient.POINts

Description

This command returns the size of array of calibration coefficient data of Impedance measurement which is retuned by SCPI.SENSe(Ch).Z.CORRection.COEFficient.DATA

Variable

Parameter	*Value*
Description	size of array
Data Type	Long integer type (Long)
Range	0 to 1601
Preset Value	0
Unit	-
Resolution	-

Examples

See SCPI.SENSe(Ch).Z.CORRection.COEFficient.DATA

Related Objects

SCPI.SENSe(Ch).Z.CORRection.COEFficient.DATA

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:CORRection:COEFficient:POINts?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:CORR:COEF:POINt?"
20 ENTER 717;A
```

SCPI.SENSe(*Ch*).Z.CORRection.COLLect.ACQuire

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).Z.CORRection.COLLect.ACQuire = *Standard*

Description

This command executes the open, short, load and low-loss capacitor of the impedance calibration for the selected channel (*Ch*).

Variable

Parameter	*Standard*
Description	Standard Type
Data Type	Character string type (String)
Range	OPEN: Open SHORt: Short LOAD: Load LOAD2: Low- Loss capacitor
Preset Value	-
Unit	-
Resolution	-

Examples

```
Dim Dmy As Long
MsgBox "Connect Open"
SCPI.SENSe.Z.CORRection.COLLect.ACQuire = "OPEN"
Dmy = SCPI.IEEE4882.OPC
MsgBox "Connect Short"
SCPI.SENSe.Z.CORRection.COLLect.ACQuire = "SHORT"
Dmy = SCPI.IEEE4882.OPC
MsgBox "Connect Load"
SCPI.SENSe.Z.CORRection.COLLect.ACQuire = "LOAD"
Dmy = SCPI.IEEE4882.OPC
```

SCPI.SENSe.Z.CORRection.COLLect.SAVE

Related Objects

SCPI.SENSe(Ch).Z.CORRection.COLLect.SAVE

Equivalent Key

Calibration > Calibrate > Impedance Calibration > Open [Open(f)]

Calibration > Calibrate > Impedance Calibration > Short [Short(f)]

Calibration > Calibrate > Impedance Calibration > Load [Broadband]

Calibration > Calibrate > Impedance Calibration > Low-Loss C (optional)

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:CORRection:COLLect[:ACQuire]
{OPEN|SHORt|LOAD|LOAD2}

Example of use

```
10 OUTPUT 717;":SENS1:Z:CORR:COLL OPEN"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
40 OUTPUT 717;":SENS1:Z:CORR:COLL SHORT"
50 OUTPUT 717;"*OPC?"
60 ENTER 717;A
70 OUTPUT 717;":SENS1:Z:CORR:COLL LOAD"
80 OUTPUT 717;"*OPC?"
90 ENTER 717;A
100 OUTPUT 717;":SENS1:Z:CORR:COLL:SAVE"
```

SCPI.SENSe(*Ch*).Z.CORRection.COLLect.SAVE

Object Type

Method (**Write Only**)

Syntax

SCPI.SENSe(*Ch*).Z.CORRection.COLLect.SAVE

Description

This command calculates the calibration coefficients for impedance calibration for the selected channel (*Ch*).

Examples

See SCPI.SENSe(Ch).Z.CORRection.COLLect.ACQuire

Related Objects

SCPI.SENSe(Ch).Z.CORRection.COLLect.ACQuire

Equivalent Key

Calibration > **Calibrate** > **Impedance Calibration** > **Done**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:CORRection:COLLect:SAVE

Example of use

See :SENSe{[1]-4}:Z:CORRection:COLLect[:ACQuire]

SCPI.SENSe(*Ch*).Z.METHod

Object Type

Property (**Read-Write**)

Syntax

SCPI.SENSe(*Ch*).Z.METHod = *Value*

Value = SCPI.SENSe(*Ch*).Z.METHod

Description

This command sets or gets the impedance method.

When the method changes, the Fixture Compensation table is cleared.

Variable

Parameter	*Value*
Description	Impedance method
Data Type	Character string type (String)
Range	Select one of the following options: P1Reflection:S-Parameter Port 1 reflection measurement (for simple impedance measurement) P2Reflection:S-Parameter Port 2 reflection measurement (for simple impedance measurement) TSERies:S-Parameter series-through measurement (for simple impedance measurement) TSHunt:S-Parameter shunt-through measurement (for PDN component characterization) GSERies:Gain-Phase series-through measurement (for simple impedance measurement) GSHunt:Gain-Phase shunt-through measurement (for PDN component characterization)
Preset Value	P1Reflection
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var= "P1Reflection"
SCPI.SENSe(1).Z.METHod = Var
Var = SCPI.SENSe(1).Z.METHod
```

Equivalent Key

There is no equivalent key is available on the front panel. However, the similar key is:

Meas > **Method** > **Port 1 Refl** OR

Meas > **Method** > **Port 2 Refl** OR

Meas > **Method** > **Port 1-2 Series** OR

Meas > **Method** > **Port 1-2 Shunt** OR

Meas > **Method** > **GP Series T 50Ω R 1MΩ** OR

Meas > **Method** > **GP Shunt T 50Ω R 50Ω**

Equivalent SCPI Command

Syntax

:SENSe{[1]-4}:Z:METHod <string>

:SENSe{[1]-4}:Z:METHod?

Query Response

<string><newline><^END>

Example of use

```
10 OUTPUT 717;":SENS1:Z:METH P1R"
20 OUTPUT 717;":SENS1:Z:METH?"
30 ENTER 717;A$
```

SERVICE

SCPI.SERVice.ACHannel.ACTive

Object type

Property (**Read-Write**)

Syntax

SCPI.SERVice.ACHannel.ACTive = *Value*

Value = SCPI.SERVice.ACHannel.ACTive

Description

This command sets/gets the active channel number.

Variable

Parameter	*Value*
Description	Active channel number
Data type	Long integer type (Long)
Range	1 to 4

Examples

```
Dim ActChan As Long
ActChan = SCPI.SERVice.ACHannel.ACTive
SCPI.SERVice.ACHannel.ACTive = 2
```

Related objects

SCPI.SERVice.ACHannel.COUNt

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SERVice:ACHannel:ACTive <numeric>

:SERVice:ACHannel:ACTive?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:ACH:ACT 2"
20 OUTPUT 717;":SERV:ACH:ACT?"
30 ENTER 717;A
```

SCPI.SERVice.ACHannel.COUNt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.ACHannel.COUNt

Description

This command reads the upper limit of the number of channels of the E5061B.

Variable

Parameter	*Value*
Description	Upper limit of the number of channels
Data type	Long integer type (Long)
Range	4

Examples

```
Dim MaxChan As Long
MaxChan = SCPI.SERVice.ACHannel.COUNt
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SERVice:ACHannel:COUNt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:ACH:COUN?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel(Ch).ATRace.ACTive

Object Type

Property (**Read-Write**)

Syntax

SCPI.SERVice.CHANnel(*Ch*).ATRace.ACTive = *Value*

Value = SCPI.SERVice.CHANnel(*Ch*).ATRace.ACTive

Description

This command sets/gets the active trace number of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Active trace number
Data Type	Long integer type (Long)
Range	1 to 4

Examples

```
Dim NumTrac as Long
NumTrac = SCPI.SERVice.CHANnel(4).ATRace.ACTive
SCPI.SERVice.CHANnel(4).ATRace.ACTive = 3
```

Related Objects

SCPI.SERVice.CHANnel(Ch).ATRace.COUNt

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:ATRace:ACTive?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:ATR:ACT 2"
20 OUTPUT 717;":SERV:CHAN2:ATR:ACT?"
30 ENTER 717;A
```

SCPI.SERVice.CHANnel(Ch).ATRace.COUNt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.CHANnel(*Ch*).ATRace.COUNt

Description

This command reads the upper limit of the number of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Upper limit of the number of traces.
Data type	Long integer type (Long)
Range	4

Examples

```
Dim MaxTrac As Long
MaxTrac = SCPI.SERVice.CHANnel(1).ATRace.COUNt
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SERVice:CHANnel{[1]-4}:ATRace:COUNt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN2:ATR:COUN?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.BIAS

Object Type

Property (**Read-Write**)

Syntax

SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.BIAS = *Value*

Value = SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.BIAS

Description

This command sets or gets the X-Axis DC Bias annotation mode display to Start/Stop or Center/Span.

Variable

Parameter	*Value*
Description	Display of Start/Stop or Center/Span value at the bottom of the screen.
Data Type	Character string type (String)
Range	Select from either of the following: "SSTop": Displays the Start and Stop value. "CSPan": Displays the Centre and Span value.
Preset Value	SSTop
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var= "SSTop"
SCPI.SERVice.CHANnel(1).DISPlay.ANNotation.XAXis.MODE.BIAS = Var
Var = SCPI.SERVice.CHANnel(1).DISPlay.ANNotation.XAXis.MODE.BIAS
```

Related Objects

SCPI.SERVice.CHANnel(Ch).DISPlay.ANNotation.XAXis.MODE.FREQuency

SCPI.SERVice.CHANnel(Ch).DISPlay.ANNotation.XAXis.MODE.POWer

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:DISPlay:ANNotation:XAXis:MODE:BIAS {SSTop|CSPan}

:SERVice:CHANnel{[1]-4}:DISPlay:ANNotation:XAXis:MODE:BIAS?

Query Response

{SST|CSP}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:DISP:ANN:XAX:MODE:BIAS SSTop"
20 OUTPUT 717;":SERV:CHAN1:DISP:ANN:XAX:MODE:BIAS?"
30 ENTER 717;A$
```

SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.FREQuency

Object Type

Property (**Read-Write**)

Syntax

SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.FREQuency = *Value*

Value = SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.FREQuency

Description

This command sets or gets X-Axis frequency annotation mode display to Start/Stop or Center/Span.

Variable

Parameter	*Value*
Description	Display of Start/Stop or Center/Span value at the bottom of the screen.
Data Type	Character string type (String)
Range	Select from either of the following: "SSTop": Displays the Start and Stop value. "CSPan": Displays the Centre and Span value.
Preset Value	SSTop
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var= "SSTop"
SCPI.SERVice.CHANnel(1).DISPlay.ANNotation.XAXis.MODE.FREQuency = Var
Var = SCPI.SERVice.CHANnel(1).DISPlay.ANNotation.XAXis.MODE.FREQuency
```

Related Objects

SCPI.SERVice.CHANnel(Ch).DISPlay.ANNotation.XAXis.MODE.BIAS

SCPI.SERVice.CHANnel(Ch).DISPlay.ANNotation.XAXis.MODE.POWer

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:DISPlay:ANNotation:XAXis:MODE:FREQuency {SSTop|CSPan}

:SERVice:CHANnel{[1]-4}:DISPlay:ANNotation:XAXis:MODE:FREQuency?

Query Response

{SST|CSP}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:DISP:ANN:XAX:MODE:FREQ SSTop"
20 OUTPUT 717;":SERV:CHAN1:DISP:ANN:XAX:MODE:FREQ?"
30 ENTER 717;A$
```

SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.POWer

Object Type

Property (**Read-Write**)

Syntax

SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.POWer = *Value*

Value = SCPI.SERVice.CHANnel(*Ch*).DISPlay.ANNotation.XAXis.MODE.POWer

Description

This command sets or gets X-Axis power annotation mode display to Start/Stop or Center/Span.

Variable

Parameter	*Value*
Description	Display of Start/Stop or Center/Span value at the bottom of the screen.
Data Type	Character string type (String)
Range	Select from either of the following: "SSTop": Displays the Start and Stop value. "CSPan": Displays the Centre and Span value.
Preset Value	SSTop
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var= "SSTop"
SCPI.SERVice.CHANnel(1).DISPlay.ANNotation.XAXis.MODE.POWer = Var
Var = SCPI.SERVice.CHANnel(1).DISPlay.ANNotation.XAXis.MODE.POWer
```

Related Objects

SCPI.SERVice.CHANnel(Ch).DISPlay.ANNotation.XAXis.MODE.FREQuency

SCPI.SERVice.CHANnel(Ch).DISPlay.ANNotation.XAXis.MODE.POWer

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:DISPlay:ANNotation:XAXis:MODE:POWer {SSTop|CSPan}

:SERVice:CHANnel{[1]-4}:DISPlay:ANNotation:XAXis:MODE:POWer?

Query Response

{SST|CSP}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:DISP:ANN:XAX:MODE:POW SST"
20 OUTPUT 717;":SERV:CHANnel1:DISPlay:ANN:XAX:MODE:POW?"
30 ENTER 717;A$
```

SCPI.SERVice.CHANnel(*Ch*).SEGMent.DATA

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.CHANnel(*Ch*).SEGMent.DATA

Description

This command gets number of the segment data, for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of the segment data
Data Type	Long integer type (Long)
Range	9 to 1413
Preset Value	9

Examples

```
Dim NumSegm as Long
NumSegm = SCPI.SERVice.CHANnel(2).SEGMent.DATA
```

Related Objects

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:SEGMent:DATA?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN3:SEGM:DATA?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel(*Ch*).STAN(*Std*).TABLe

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.CHANnel(*Ch*).STAN(*Std*).TABLe

Description

This command returns the number of point of the table data which is set by SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).TABLe.

Variable

Parameter	*Value*
Description	Number of standard table data
Data Type	Long integer type (Long)
Range	1 to 4804
Preset Value	1
Unit	-
Resolution	-

Examples

See SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).TABLe

Related objects

SCPI.SENSe(Ch).CORRection.COLLect.CKIT.STAN(Std).TABLe

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:STAN{[1]-21}:TABLe?

Query Response

<numeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:STAN1:TABL?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel(*Ch*).SWEep.POINts

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.CHANnel(*Ch*).SWEep.POINts

Description

This command gets the number of measurement points of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of measurement points
Data Type	Long integer type (Long)

Examples

```
Dim Poin as Long
Poin = SCPI.SERVice.CHANnel(2).SWEep.POINts
```

Related Objects

SCPI.SENSe(Ch).SWEep.POINts

SCPI.SENSe(Ch).SEGMent.SWEep.POINts

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:SWEep:POINts?

Query Response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN2:SWE:POIN?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel(Ch).TRACe(Tr).AMRKer.ACTive

Object Type

Property (**Read-Write**)

Syntax

SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).AMRKer.ACTive = *Value*

Value = SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).AMRKer.ACTive

Description

This command sets/gets the active marker number of selected trace (*Tr*) in the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Active marker number of selected trace in selected channel
Data Type	Long integer type (Long)
Range	0 to 10

Examples

```
Dim ActMkr as Long
ActMkr = SCPI.SERVice.CHANnel(4).TRACe.AMRKer.ACTive
SCPI.SERVice.CHANnel(4).TRACe.AMRKer.ACTive = 6
```

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:TRACe{[1]-4}:AMRKer:ACTive?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN2:TRAC2:AMRK:ACT?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel(*Ch*).TRACe(Tr).DISPlay.ANNotation.XAXis.MODE.TRANsform

Object Type

Property (**Read-Write**)

Syntax

SCPI.SERVice.CHANnel(*Ch*).TRACe(Tr).DISPlay.ANNotation.XAXis.MODE.TRANsform = *Value*

Value = SCPI.SERVice.CHANnel(*Ch*).TRACe(Tr).DISPlay.ANNotation.XAXis.MODE.TRANsform

Description

This command sets or gets X-Axis annotation mode display of fault location measurements to Start/Stop or Center/Span.

Variable

Parameter	*Value*
Description	Display of Start/Stop or Center/Span value at the bottom of the screen.
Data Type	Character string type (String)
Range	Select from either of the following: "SSTop": Displays the Start and Stop value. "CSPan": Displays the Centre and Span value.
Preset Value	SSTop
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var= "SSTop"
SCPI.SERVice.CHANnel(1).TRACe(1).DISPlay.ANNotation.XAXis.MODE.TRANsform = Var
Var = SCPI.SERVice.CHANnel(1).TRACe(1).DISPlay.ANNotation.XAXis.MODE.TRANsform
```

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:TRACe{[1]-4}:DISPlay:ANNotation:XAXis:MODE:TRANsform {SSTop|CSPan}

:SERVice:CHANnel{[1]-4}:TRACe{[1]-4}:DISPlay:ANNotation:XAXis:MODE:TRANsform?

Query Response

{SST|CSP}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:TRAC1:DISP:ANN:XAX:MODE:TRAN SST"
20 OUTPUT 717;":SERV:CHAN1:TRACe1:DISP:ANN:XAX:MODE:TRAN?"
30 ENTER 717;A$
```

SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).LIMit.DATA

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).LIMit.DATA

Description

This command gets the number of array elements using SCPI.CALCulate(Ch).SELected.LIMit.DATA command, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	number of array elements
Data Type	Long integer type (Long)
Range	1 to 501

Examples

```
Dim NumLimTbl as Long
NumLimTbl = SCPI.SERVice.CHANnel(2).TRACe(4).LIMit.DATA
```

Related Objects

SCPI.CALCulate(Ch).SELected.LIMit.DATA

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:TRACe{[1]-4}:LIMit:DATA?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:TRAC3:LIM:DATA?"
20 ENTER 717;A
```

SCPI.SERVice.CHANnel(Ch).TRACe(Tr).MEMValid

Object Type

Property (**Read-Write**)

Syntax

SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).MEMValid = *Status*

Status = SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).MEMValid

Description

This command turns ON/OFF the memory trace for the selected trace (*Tr*) of the selected channel (*Ch*). The memory trace can be turned on even if it is inactivated (no data).

NOTE This command does not copy the data trace to memory trace. When the data trace copies to the memory trace, use the SCPI.CALCulate(Ch).SELected.MATH.MEMorize command.

Variable

Parameter	*Status*
Description	ON/OFF the memory trace
Data Type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the memory trace • False or OFF: Turns OFF the memory trace
Preset Value	False or OFF

Examples

```
Dim StatMemTr as Boolean
SCPI.SERVice.CHANnel(4).TRACe(2).MEMValid = True
StatMemTr = SCPI.SERVice.CHANnel(4).TRACe(2).MEMValid
```

Related Objects

SCPI.CALCulate(Ch).SELected.MATH.MEMorize

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:TRACe{[1]-4}:MEMValid {ON|OFF|1|0}
:SERVice:CHANnel{[1]-4}:TRACe{[1]-4}:MEMValid?

Query Response

{1|0} <newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:TRAC2:MEMV ON"
20 OUTPUT 717;":SERV:CHAN1:TRAC3:MEMV?"
30 ENTER 717;A
```

SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).RLIMit.DATA

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.CHANnel(*Ch*).TRACe(*Tr*).RLIMit.DATA

Description

This command gets the number of array elements using SCPI.CALCulate(Ch).SELected.RLIMit.DATA command, for the selected trace (*Tr*) of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Number of array elements
Data Type	Long integer type (Long)
Range	1 to 49

Examples

```
Dim NumRlimTbl as Long
NumRlimTbl = SCPI.SERVice.CHANnel(1).TRACe(2).RLIMit.DATA
```

Related Objects

SCPI.CALCulate(Ch).SELected.RLIMit.DATA

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:CHANnel{[1]-4}:TRACe{[1]-4}:RLIMit:DATA?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:CHAN1:TRAC3:RLIM:DATA?"
20 ENTER 717;A
```

SCPI.SERVice.LOGGing.CLEar

Object Type

Method (**Write Only**)

Syntax

SCPI.SERVice.LOGGing.CLEar

Description

This command clears all log files about Event, Power on test, Mech software, Overload, FW close, Recovery.

Examples

SCPI.SERVice.LOGGing.CLEar

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:LOGGing:CLEar

Example of use

```
10 OUTPUT 717;":SERV:LOGG:CLE"
```

SCPI.SERVice.POWer.OVERload.PROTect.CLEar

Object type

Method (**Write Only**)

Syntax

SCPI.SERVice.POWer.OVERload.PROTect.CLEar

Description

This command clears the overload protection.

Examples

SCPI.SERVice.POWer.OVERload.PROTect.CLEar

Related objects

SCPI.SERVice.POWer.OVERload.PROTect.STATe

Equivalent key

System > **Overload Recovery** > **Clear Overload Protection**

Equivalent SCPI command

Syntax

:SERVice:POWer:OVERload:PROTect:CLEar

Example of use

```
10 OUTPUT 717;":SERV:POW:OVER:PROT:CLE"
```

SCPI.SERVice.POWer.OVERload.PROTect.STATe

Object Type

Property (**Read Only**)

Syntax

Status = SCPI.SERVice.POWer.OVERload.PROTect.STATe

Description

This command queries if the overload protection is activated (Overload protection indicator is displayed).

Variable

Parameter	*Status*
Description	Status of overload protection
Data type	Boolean type (Boolean)
Range	Return from either of the following: • True: overload protection is activated. • False: overload protection is off.
Preset value	False

Examples

```
Dim OverLoadStatus As Boolean
OverLoadStatus = SCPI.SERVice.POWer.OVERload.PROTect.STATe
```

Related Objects

SCPI.SERVice.POWer.OVERload.PROTect.CLEar

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:POWer:OVERload:PROTect:STATe?

Query Response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:POW:OVER:PROT:STAT?"
20 ENTER 717;A
```

SCPI.SERVice.SREVision

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SREVision

Description

This command reads the system version (OS ID) of the E5061B.

Variable

Parameter	*Value*
Description	1: Windows Vista Business (downgraded to XP) 0: Windoes XP
Data type	Long integer type (Long)

Examples

```
Dim SystemRevision As Long
SystemRevision = SCPI.SERVice.SREVision
```

Equivalent key

System > **Firmware Revision**

OS ID shows this status.

Equivalent SCPI command

Syntax

:SERVice:SREVision?

Query response

<newmeric><newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SREV?"
20 ENTER 717;A
```

SCPI.SERVice.SWEep.FREQuency.MAXimum

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SWEep.FREQuency.MAXimum

Description

This command reads the upper limit of measurement frequency of E5061B.

Variable

Parameter	*Value*
Description	Upper limit of measurement frequency
Data type	Double precision floating point type (Double)

Examples

```
Dim MaxFreq As Double
MaxFreq = SCPI.SERVice.SWEep.FREQuency.MAXimum
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SERVice:SWEep:FREQuency:MAXimum?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SWE:FREQ:MAX?"
20 ENTER 717;A
```

SCPI.SERVice.SWEep.FREQuency.MINimum

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SWEep.FREQuency.MINimum

Description

This command reads the lower limit of measurement frequency of E5061B.

Variable

Parameter	*Value*
Description	Lower limit of measurement frequency
Data type	Double precision floating point type (Double)

Examples

```
Dim MinFreq As Double
MinFreq = SCPI.SERVice.SWEep.FREQuency.MINimum
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SERVice:SWEep:FREQuency:MINimum?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SWE:FREQ:MIN?"
20 ENTER 717;A
```

SCPI.SERVice.SWEep.POINts

Object type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SWEep.POINts

Description

This command reads the upper limit of the number of measurement points at the current configuration.

Variable

Parameter	*Value*
Description	Upper limit of the number of points
Data type	Long integer type (Long)

Examples

```
Dim MaxPoin As Long
MaxPoin = SCPI.SERVice.SWEep.POINts
```

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SERVice:SWEep:POINts?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SWE:POIN?"
20 ENTER 717;A
```

SCPI.SERVice.SYSTem.OPTion.GPHase

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SYSTem.OPTion.GPHase

Description

This query command gets the availability of the Gain-Phase test port option.

Variable

Parameter	*Status*
Description	Gets the availability of the Gain-Phase test port option
Data Type	Boolean type (Boolean)
Range	The return values are: True: The Gain-Phase test port option is available. False: The Gain-Phase test port option is not available.
Preset Value	-
Unit	-
Resolution	-

Examples

```
Dim Var as Boolean
Var = SCPI.SERVice.SYSTem.OPTion.GPHase
```

Related Objects

SCPI.SERVice.SYSTem.OPTion.TPIMpedance

SCPI.SERVice.SYSTem.OPTion.TSET

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:SYSTem:OPTion:GPHase?

Query response

1910

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SYST:OPT:GPH?"
20 ENTER 717;A
```

SCPI.SERVice.SYSTem.OPTion.TPIMpedance

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SYSTem.OPTion.TPIMpedance

Description

This query command gets the test port impedance option, either 50 or 75.

Variable

Parameter	*Value*
Description	Test port impedance
Data Type	Long integer type (Long)
Range	The return values are: 50: The value of test port impedance option is 50 Ohm. 75: The value of test port impedance option is 75 Ohm.
Preset Value	-
Unit	Ohm (Ω)
Resolution	-

Examples

```
Dim Var as Long
Var = SCPI.SERVice.SYSTem.OPTion.TPIMpedance
```

Related Objects

SCPI.SERVice.SYSTem.OPTion.GPHase

SCPI.SERVice.SYSTem.OPTion.TSET

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:SYSTem:OPTion:TPIMpedance?

Query Response

{50|75}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SYST:OPT:TPIM?"
20 ENTER 717;A
```

SCPI.SERVice.SYSTem.OPTion.TSET

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SYSTem.OPTion.TSET

Description

This query command gets the value of the test set option, either S-Parameter or T/R.

Variable

Parameter	*Value*
Description	Test set option
Data Type	Character string type (String)
Range	The return values are: SPARAM: The value of test set option is S-Parameter. TR: The value of test set option is Transmission/Reflection.
Preset Value	-
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var = SCPI.SERVice.SYSTem.OPTion.TSET
```

Related Objects

SCPI.SERVice.SYSTem.OPTion.GPHase

SCPI.SERVice.SYSTem.OPTion.TPIMpedance

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SERVice:SYSTem:OPTion:TSET?

Query Response

{SPARM|TR}><newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SYST:OPT:TSET?"
20 ENTER 717;A$
```

SCPI.SERVice.SYSTem.OS.REVision

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SERVice.SYSTem.OS.REVision

Description

This command gets the HDD Image revision. Returns 'EL11x' for EL100/EL110/112/114.

Variable

Parameter	*Value*
Description	Shows HDD image revision
Data Type	Character string type (String)
Range	254 chars
Preset Value	-
Unit	-
Resolution	-

Examples

```
Dim Var as String
Var = SCPI.SERVice.SYSTem.OS.REVision
```

Equivalent Key

System > **Firmware Revision**

HDD Image shows this status.

Equivalent SCPI Command

Syntax

:SERVice:SYSTem:OS:REVision?

Query Response

{String}<newline><^END>

Example of use

```
10 OUTPUT 717;":SERV:SYST:OS:REV?"
20 ENTER 717;A$
```

SOURCE

SCPI.SOURce(*Ch*).BIAS.CENTer

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).BIAS.CENTer = *Value*

Value = SCPI.SOURce(*Ch*).BIAS.CENTer

Description

This command sets/gets the center value of the bias sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Center voltage
Data Type	Double precision floating point type (Double)
Range	-40 to 40
Preset Value	0
Unit	V (voltage)
Unit	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim CentVolt as Double
SCPI.SOURce(4).BIAS.CENTer = 3.0
CentVolt = SCPI.SOURce(4).BIAS.CENTer
```

Related Objects

SCPI.SOURce(Ch).BIAS.SPAN

SCPI.SOURce(Ch).BIAS.STARt

SCPI.SOURce(Ch).BIAS.STOP

Equivalent Key

Center

Equivalent SCPI Command

Syntax

```
:SOURce{[1]-4}:BIAS:CENTer <numeric>
:SOURce{[1]-4}:BIAS:CENTer?
```

Query Response

```
{numeric} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR1:BIAS:CENT 1.2"
20 OUTPUT 717;":SOUR1:BIAS:CENT?"
30 ENTER 717;A
```

SCPI.SOURce(Ch).BIAS.RANGe.AUTO

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).BIAS.RANGe.AUTO = *Status*

Status = SCPI.SOURce(*Ch*).BIAS.RANGe.AUTO

Description

This command sets/gets to fix the dc bias range at 40 V range for the selected channel (*Ch*).

Variable

Parameter	*Status*
Description	ON/OFF of the auto setting of the sweep time
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Auto. • False or OFF: Range is fixed at 40 V range.
Preset value	True or ON

Examples

```
Dim DCBiasRange As Boolean
SCPI.SOURce(1).BIAS.RANGe.AUTO = False
DCBiasRange = SCPI.SOURce(1).BIAS.RANGe.AUTO
```

Equivalent key

System > **Service Menu** > **DC Bias Range**

Equivalent SCPI command

Syntax

:SOURce{[1]-4}:BIAS:RANGe:AUTO {ON|OFF|1|0}

:SOURce{[1]-4}:BIAS:RANGe:AUTO?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:BIAS:RANG:AUTO OFF"
20 OUTPUT 717;":SOUR1:BIAS:RANG:AUTO?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).BIAS.SPAN

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).BIAS.SPAN = *Value*

Value = SCPI.SOURce(*Ch*).BIAS.SPAN

Description

This command sets/gets the span value of the bias sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Span voltage
Data Type	Double precision floating point type (Double)
Range	0 to 80
Preset Value	2
Unit	V (voltage)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim SpanVolt as Double
SCPI.SOURce(3).BIAS.SPAN = 4.0
SpanVolt = SCPI.SOURce(3).BIAS.SPAN
```

Related Objects

SCPI.SOURce(Ch).BIAS.CENTer

SCPI.SOURce(Ch).BIAS.STARt

SCPI.SOURce(Ch).BIAS.STOP

Equivalent Key

Span

Equivalent SCPI Command

Syntax

```
:SOURce{[1]-4}:BIAS:SPAN <numeric>
:SOURce{[1]-4}:BIAS:SPAN?
```

Query Response

```
{numeric} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR1:BIAS:SPAN 2.0"
20 OUTPUT 717;":SOUR1:BIAS:SPAN?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).BIAS.STARt

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).BIAS.STARt = *Value*

Value = SCPI.SOURce(*Ch*).BIAS.STARt

Description

This command sets/gets the start value of the bias sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Start voltage
Data Type	Double precision floating point type (Double)
Range	-40 to 40
Preset Value	-1
Unit	V (voltage)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StarVolt as Double
SCPI.SOURce(1).BIAS.STARt = -1.5
StarVolt = SCPI.SOURce(1).BIAS.STARt
```

Related Objects

SCPI.SOURce(Ch).BIAS.CENTer

SCPI.SOURce(Ch).BIAS.SPAN

SCPI.SOURce(Ch).BIAS.STOP

Equivalent Key

Start

Equivalent SCPI Command

Syntax

```
:SOURce{[1]-4}:BIAS:STARt <numeric>
:SOURce{[1]-4}:BIAS:STARt?
```

Query Response

```
{numeric} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR2:BIAS:STAR -12.0"
20 OUTPUT 717;":SOUR2:BIAS:STAR?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).BIAS.STOP

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).BIAS.STOP = *Value*

Value = SCPI.SOURce(*Ch*).BIAS.STOP

Description

This command sets/gets the stop value of the bias sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Stop voltage
Data Type	Double precision floating point type (Double)
Range	-40 to 40
Preset Value	1
Unit	V (voltage)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim StopVolt as Double
SCPI.SOURce(1).BIAS.STOP = 2.5
StopVolt = SCPI.SOURce(1).BIAS.STOP
```

Related Objects

SCPI.SOURce(Ch).BIAS.CENTer

SCPI.SOURce(Ch).BIAS.SPAN

SCPI.SOURce(Ch).BIAS.STARt

Equivalent Key

Stop

Equivalent SCPI Command

Syntax

```
:SOURce{[1]-4}:BIAS:STOP <numeric>
:SOURce{[1]-4}:BIAS:STOP?
```

Query Response

```
{numeric} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR2:BIAS:STOP 5.0"
20 OUTPUT 717;":SOUR2:BIAS:STOP?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.ATTenuation.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.ATTenuation.DATA = *Value*

Value = SCPI.SOURce(*Ch*).POWer.ATTenuation.DATA

Description

This command does nothing. The E5062B has no attenuator setting for power. This command is provided because of its command compatibility with E5061A/E5062B.

Variable

Parameter	*Value*
Description	Dummy Parameter
Data type	Long integer type (Long)
Range	0 to 40
Preset value	0
Unit	dB
Resolution	10

Examples

```
Dim Att As Long
SCPI.SOURce(1).POWer.ATTenuation.DATA = 10
Att = SCPI.SOURce(1).POWer.ATTenuation.DATA
```

Related objects

None.

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SOURce{[1]-4}:POWer:ATTenuation[:DATA] <numeric>

:SOURce{[1]-4}:POWer:ATTenuation[:DATA]?

Query response

0<newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:POW:ATT.DATA 20"
20 OUTPUT 717;":SOUR1:POW:ATT.DATA?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.CENTer

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.CENTer = *Value*

Value = SCPI.SOURce(*Ch*).POWer.CENTer

Description

This command sets/gets the center value of the power sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Center value
Data type	Double precision floating point type (Double)
Range	–45 to 10 dBm
Preset value	-2.5
Unit	dBm
Resolution	0.05 or 0.025
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Pcntr As Double
SCPI.SOURce(1).POWer.CENTer = 0
Pcntr = SCPI.SOURce(1).POWer.CENTer
```

Related objects

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SOURce(Ch).POWer.SPAN

Equivalent key

Center

Equivalent SCPI command

Syntax

:SOURce{[1]-4}:POWer:CENTer <numeric>

:SOURce{[1]-4}:POWer:CENTer?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:POW:CENT 0"
20 OUTPUT 717;":SOUR1:POW:CENT?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.GPPort.LEVel.IMMediate.AMPLitude

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.GPPort.LEVel.IMMediate.AMPLitude = *Value*

Value = SCPI.SOURce(*Ch*).POWer.GPPort.LEVel.IMMediate.AMPLitude

Description

This command sets/gets the power level for gain phase measurement (LF out) for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Power level
Data Type	Double precision floating point type (Double)
Range	-45 to 10 dBm
Preset Value	0
Unit	dBm
Resolution	0.05
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim GPLev as Double
SCPI.SOURce(2).POWer.GPPort.LEVel.IMMediate.AMPLitude = -15.0
GPLev = SCPI.SOURce(1).POWer.GPPort.LEVel.IMMediate.AMPLitude
```

Related Objects

SCPI.SOURce(Ch).POWer.PORT(Pt).LEVel.IMMediate.AMPLitude

SCPI.OUTPut.STATe

Equivalent Key

Sweep Setup > **Power** > **Port Power** > **LF OUT Power**

Equivalent SCPI Command

Syntax

:SOURce{[1]-4}:POWer:GPPort[:LEVel][:IMMediate][:AMPLitude] <numeric>

:SOURce{[1]-4}:POWer:GPPort[:LEVel][:IMMediate][:AMPLitude]?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:POW:GPP -5.0"
20 OUTPUT 717;":SOUR1:POW:GPP?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.LEVel.IMMediate.AMPLitude

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.LEVel.IMMediate.AMPLitude = *Value*

Value = SCPI.SOURce(*Ch*).POWer.LEVel.IMMediate.AMPLitude

Description

This command sets/gets the power level of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Power level
Data type	Double precision floating point type (Double)
Range	-45 to 10 dBm
Preset value	0
Unit	dBm
Resolution	0.05
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PowLev As Double
SCPI.SOURce(1).POWer.LEVel.IMMediate.AMPLitude = -10
PowLev = SCPI.SOURce(1).POWer.LEVel.IMMediate.AMPLitude
```

Related objects

SCPI.SOURce(Ch).POWer.GPPort.LEVel.IMMediate.AMPLitude

SCPI.OUTPut.STATe

Equivalent key

Sweep Setup > Power > Power

Equivalent SCPI command

Syntax

:SOURce{[1]-4}:POWer[:LEVel][:IMMediate][:AMPLitude] <numeric>
:SOURce{[1]-4}:POWer[:LEVel][:IMMediate][:AMPLitude]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:POW -12.5"
20 OUTPUT 717;":SOUR1:POW?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.DATA

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.DATA = *Value*

Value = SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.DATA

Description

This command sets/gets the correction value of the power slope feature of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Correction value of the power slope feature
Data type	Double precision floating point type (Double)
Range	-2 to 2
Preset value	0
Unit	dB/GHz
Resolution	0.01
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim SlopLev As Double
SCPI.SOURce(1).POWer.LEVel.SLOPe.DATA = 0.1
SlopLev = SCPI.SOURce(1).POWer.LEVel.SLOPe.DATA
```

Related objects

SCPI.SOURce(Ch).POWer.LEVel.SLOPe.STATe

Equivalent key

Sweep Setup > **Power** > **Slope**

Equivalent SCPI command

Syntax

```
:SOURce{[1]-4}:POWer[:LEVel]:SLOPe[:DATA] <numeric>
:SOURce{[1]-4}:POWer[:LEVel]:SLOPe[:DATA]?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR1:POW:SLOP 0.1"
20 OUTPUT 717;":SOUR1:POW:SLOP?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.STATe = *Status*

Status = SCPI.SOURce(*Ch*).POWer.LEVel.SLOPe.STATe

Description

This command turns ON/OFF or gets the status of the power slope feature, for the selected channel (*Ch*). This command corrects the attenuation of simple power level proportional to the frequency (attenuation due to cables etc).

Variable

Parameter	*Status*
Description	On/off of the power slope feature
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns on the power slop feature. • False or OFF: Turns off the power slop feature.
Preset value	False or OFF

Examples

```
Dim Slop As Boolean
SCPI.SOURce(1).POWer.LEVel.SLOPe.STATe = True
Slop = SCPI.SOURce(1).POWer.LEVel.SLOPe.STATe
```

Related objects

SCPI.SOURce(Ch).POWer.LEVel.SLOPe.DATA

Equivalent key

Sweep Setup > **Power** > **Slope**

Equivalent SCPI command

Syntax

:SOURce{[1]-4}:POWer[:LEVel]:SLOPe:STATe {ON|OFF|1|0}

:SOURce{[1]-4}:POWer[:LEVel]:SLOPe:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:POW:SLOP:STAT ON"
20 OUTPUT 717;":SOUR1:POW:SLOP:STAT?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).LEVel.IMMediate.AMPLitude

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).LEVel.IMMediate.AMPLitude = *Value*

Value = SCPI.SOURce(*Ch*).POWer.PORT(*Pt*).LEVel.IMMediate.AMPLitude

Description

This command sets/gets the power level for the selected port (*Pt*) of the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Power level at the specified port.
Data type	Double precision floating point type (Double)
Range	-45 to 10 dBm
Preset value	0
Unit	dBm
Resolution	0.05
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim PowLev As Double
SCPI.SOURce(1).POWer.PORT.COUPle = False
SCPI.SOURce(1).POWer.PORT(1).LEVel.IMMediate.AMPLitude = -12.5
PowLev = SCPI.SOURce(1).POWer.PORT(1).LEVel.IMMediate.AMPLitude
```

Related objects

SCPI.SOURce(Ch).POWer.PORT.COUPle

SCPI.SOURce(Ch).POWer.GPPort.LEVel.IMMediate.AMPLitude

Equivalent key

Sweep Setup > Power > Port Power > Port 1 Power|Port 2 Power

Equivalent SCPI command

Syntax

```
:SOURce{[1]-4}:POWer:PORT{[1]|2}[:LEVel][:IMMediate][:AMPLitude]
<numeric>
:SOURce{[1]-4}:POWer:PORT{[1]|2}[:LEVel][:IMMediate][:AMPLitude]?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR1:POW:PORT1 -12.5"
20 OUTPUT 717;":SOUR1:POW:PORT1?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.PORT.COUPle

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.PORT.COUPle = *Status*

Status = SCPI.SOURce(*Ch*).POWer.PORT.COUPle

Description

This command sets/gets whether to output the same power level for each port of the selected channel (*Ch*).

When the power slope feature is ON, the same power level is always output to all ports regardless of this setting because different power levels cannot be output for each port.

Variable

Parameter	*Status*
Description	Turns ON/OFF the coupling between ports for the power level output
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Outputs the same power level to individual ports. • False or OFF: Outputs different power levels to individual ports.
Preset value	True or ON

Examples

```
Dim OutCpl As Boolean
SCPI.SOURce(1).POWer.PORT.COUPle = False
OutCpl = SCPI.SOURce(1).POWer.PORT.COUPle
```

Related objects

SCPI.SOURce(Ch).POWer.PORT(Pt).LEVel.IMMediate.AMPLitude

Equivalent key

Sweep Setup > **Power** > **Port Couple**

Equivalent SCPI command

Syntax

```
:SOURce{[1]-4}:POWer:PORT:COUPle {ON|OFF|1|0}
:SOURce{[1]-4}:POWer:PORT:COUPle?
```

Query response

```
{1|0}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR1:POW:PORT:COUP OFF"
20 OUTPUT 717;":SOUR1:POW:PORT:COUP?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.SPAN

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.SPAN = *Value*

Value = SCPI.SOURce(*Ch*).POWer.SPAN

Description

This command sets/gets the span value of the power sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Span value
Data type	Double precision floating point type (Double)
Range	0 to 55 dB
Preset value	15
Unit	dB
Resolution	0.05
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Pspan As Double
SCPI.SOURce(1).POWer.SPAN = 10
Pspan = SCPI.SOURce(1).POWer.SPAN
```

Related objects

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SOURce(Ch).POWer.CENTer

Equivalent key

Span

Equivalent SCPI command

Syntax

:SOURce{[1]-4}:POWer:SPAN <numeric>
:SOURce{[1]-4}:POWer:SPAN?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:POW:SPAN 10"
20 OUTPUT 717;":SOUR1:POW:SPAN?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.STARt

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.STARt = *Value*

Value = SCPI.SOURce(*Ch*).POWer.STARt

Description

This command sets/gets the start value of the power sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Start value
Data type	Double precision floating point type (Double)
Range	–45 to 10 dBm
Preset value	-5
Unit	dBm
Resolution	0.05

Examples

```
Dim Pstart As Double
SCPI.SOURce(1).POWer.STARt = -10
Pstart = SCPI.SOURce(1).POWer.STARt
```

Related objects

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SOURce(Ch).POWer.STOP

Equivalent key

Start

Equivalent SCPI command

Syntax

:SOURce{[1]-4}:POWer:STARt <numeric>

:SOURce{[1]-4}:POWer:STARt?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR1:POW:STAR -10"
20 OUTPUT 717;":SOUR1:POW:STAR?"
30 ENTER 717;A
```

SCPI.SOURce(*Ch*).POWer.STOP

Object type

Property (**Read-Write**)

Syntax

SCPI.SOURce(*Ch*).POWer.STOP = *Value*

Value = SCPI.SOURce(*Ch*).POWer.STOP

Description

This command sets/gets the stop value of the power sweep for the selected channel (*Ch*).

Variable

Parameter	*Value*
Description	Stop value
Data type	Double precision floating point type (Double)
Range	–45 to 10 dBm
Preset value	0
Unit	dBm
Resolution	0.05
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Pstop As Double
SCPI.SOURce(1).POWer.STOP = 10
Pstop = SCPI.SOURce(1).POWer.STOP
```

Related objects

SCPI.SENSe(Ch).SWEep.TYPE

SCPI.SOURce(Ch).POWer.STARt

Equivalent key

Stop

Equivalent SCPI command

Syntax

```
:SOURce{[1]-4}:POWer:STOP <numeric>
:SOURce{[1]-4}:POWer:STOP?
```

Query response

```
{numeric}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR1:POW:STOP 10"
20 OUTPUT 717;":SOUR1:POW:STOP?"
30 ENTER 717;A
```

SCPI.SOURce.BIAS.ENABle

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce.BIAS.ENABle = *Status*

Status = SCPI.SOURce.BIAS.ENABle

Description

This command turns ON/OFF the DC bias output.

Measurement cannot be made until the DC bias output is turned ON.

Variable

Parameter	*Status*
Description	ON/OFF of the stimulus signal output
Data type	Boolean type (Boolean)
Range	Select from the following: • True or ON: Turns ON the DC bias output. • False or OFF: Turns OFF the DC bias output.
Preset value	False or OFF

Examples

```
Dim StatBias as Boolean
SCPI.SOURce.BIAS.ENABle = True
StatBias = SCPI.SOURce.BIAS.ENABle
```

Related Objects

SCPI.SOURce.BIAS.PORT

SCPI.SOURce.BIAS.VOLTage

Equivalent Key

Sweep Setup > **DC Bias**

System > **Overload Recovery** > **DC Bias**

Equivalent SCPI Command

Syntax

```
:SOURce:BIAS:ENABle {ON|OFF|1|0}
:SOURce:BIAS:ENABle?
```

Query Response

```
{1|0} <newline><^END>
```

Example of use

```
10 OUTPUT 717;":SOUR:BIAS:ENAB ON"
20 OUTPUT 717;":SOUR:BIAS:ENAB?"
30 ENTER 717;A
```

SCPI.SOURce.BIAS.PORT

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce.BIAS.PORT = *Param*

Param = SCPI.SOURce.BIAS.PORT

Description

This command sets/gets the port of DC bias output. If this parameter changed, DC bias output is turned OFF.

Variable

Parameter	*Param*
Description	DC bias output port
Data Type	Character string type (String)
Range	Select from either of the following: • "LFOut": Specifies the LF output port. • "P1": Specifies the Port 1.
Preset Value	"LFOut"

Examples

```
Dim DCPort as String
SCPI.SOURce.BIAS.PORT = "LFOut"
DCPort = SCPI.SOURce.BIAS.PORT
```

Related Objects

SCPI.SOURce.BIAS.ENABle

SCPI.SOURce.BIAS.VOLTage

Equivalent Key

Sweep Setup > **DC Bias Port** > **LF Out**|**Port 1**

System > **Overload Recovery** > **DC Bias Port** > **LF Out**|**Port 1**

Equivalent SCPI Command

Syntax

:SOURce:BIAS:PORT {LFOut|P1}

:SOURce:BIAS:PORT?

Query Response

{LFO|P1} <newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR:BIAS:PORT P1"
20 OUTPUT 717;":SOUR:BIAS:PORT?"
30 ENTER 717;A$
```

SCPI.SOURce.BIAS.VOLTage

Object Type

Property (**Read-Write**)

Syntax

SCPI.SOURce.BIAS.VOLTage = *Value*

Value = SCPI.SOURce.BIAS.VOLTage

Description

This command sets/gets the output level for DC bias output.

Variable

Parameter	*Value*
Description	Output level
Data Type	Double precision floating point type (Double)
Range	-40 to 40
Preset Value	0
Unit	V (voltage)
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim DCLev as Double
SCPI.SOURce.BIAS.VOLTage = 8.5
DCLev = SCPI.SOURce.BIAS.VOLTage
```

Related Objects

SCPI.SOURce.BIAS.ENABle

SCPI.SOURce.BIAS.PORT

Equivalent Key

Sweep Setup > **DC Bias Level**

System > **Overload Recovery** > **DC Bias Level**

Equivalent SCPI Command

Syntax

:SOURce:BIAS:VOLTage <numeric>

:SOURce:BIAS:VOLTage?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SOUR:BIAS:VOLT 5.2"
20 OUTPUT 717;":SOUR:BIAS:VOLT?"
30 ENTER 717;A
```

STATUS

SCPI.STATus.OPERation.CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.OPERation.CONDition

Description

This command reads the value of the Operation Status Condition Register.

Variable

Parameter	*Value*
Description	Value of the Operation Status Condition Register
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.OPERation.CONDition
```

Related objects

SCPI.STATus.OPERation.NTRansition

SCPI.STATus.OPERation.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:OPERation:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:OPER:COND?"
20 ENTER 717;A
```

SCPI.STATus.OPERation.ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.OPERation.ENABle = *Value*

Value = SCPI.STATus.OPERation.ENABle

Description

This command sets/gets the value of Operation Status Enable Register.

Variable

Parameter	*Value*
Description	Value of the Operation Status Enable Register
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0 to 3, bit 6 to 13 and bit 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.OPERation.ENABle = 16
Stat = SCPI.STATus.OPERation.ENABle
```

Related objects

SCPI.IEEE4882.SRE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:OPERation:ENABle <numeric>

:STATus:OPERation:ENABle?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:OPER:ENAB 16"
20 OUTPUT 717;":STAT:OPER:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.OPERation.EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.OPERation.EVENt

Description

This command reads the value of the Operation Status Event Register.

Variable

Parameter	*Value*
Description	Value of the Operation Status Event Register
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.OPERation.EVENt
```

Related objects

SCPI.IEEE4882.CLS

SCPI.STATus.OPERation.NTRansition

SCPI.STATus.OPERation.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:OPERation[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:OPER?"
20 ENTER 717;A
```

SCPI.STATus.OPERation.NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.OPERation.NTRansition = *Value*

Value = SCPI.STATus.OPERation.NTRansition

Description

This command sets/gets the value of negative transition filter of the Operation Status Register.

Variable

Parameter	*Value*
Description	Value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0 to 3, bit 6 to 13 and bit 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.OPERation.NTRansition = 16
Stat = SCPI.STATus.OPERation.NTRansition
```

Related objects

SCPI.STATus.OPERation.EVENt

SCPI.STATus.OPERation.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:OPERation:NTRansition <numeric>

:STATus:OPERation:NTRansition?

Query response

1960

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:OPER:NTR 16"
20 OUTPUT 717;":STAT:OPER:NTR?"
30 ENTER 717;A
```

SCPI.STATus.OPERation.PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.OPERation.PTRansition = *Value*

Value = SCPI.STATus.OPERation.PTRansition

Description

This command sets/gets the value of positive transition filter of the Operation Status Register.

Variable

Parameter	*Value*
Description	Value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	16432
Note	The bit 0 to 3, bit 6 to 13 and bit 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.OPERation.PTRansition = 0
Stat = SCPI.STATus.OPERation.PTRansition
```

Related objects

SCPI.STATus.OPERation.EVENt

SCPI.STATus.OPERation.NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:OPERation:PTRansition <numeric>

:STATus:OPERation:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:OPER:PTR 16"
20 OUTPUT 717;":STAT:OPER:PTR?"
30 ENTER 717;A
```

SCPI.STATus.PRESet

Object type

Method (**Write Only**)

Syntax

SCPI.STATus.PRESet

Description

This command initializes all the status registers.

Examples

SCPI.STATus.PRESet

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:PRESet

Example of use

```
10 OUTPUT 717;":STAT:PRES"
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATUS.QUEStionable.BLIMit.CHANnel(*Ch*).CONDition

Description

This command reads the value of the Questionable Bandwidth Limit Channel Status Condition Register of the specified channel.

Variable

Parameter	*Value*
Description	The value of the Questionable Bandwidth Limit Channel Status Condition Register
Data type	Long integer type (Long)

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).CONDition
```

Related objects

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).NTRansition

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).ENABle = *Value*

Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).ENABle

Description

This command sets/gets the value of the "Questionable Bandwidth Limit Channel Status Enable Register" for the specified channel.

Variable

Parameter	*Value*
Description	The value of the Questionable Bandwidth Limit Channel Status Enable Register
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ENABle = 16
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).ENABle
```

Related objects

SCPI.STATus.QUEStionable.BLIMit.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}:ENABle <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}:ENABle?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ENAB 16"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).EVENt

Description

This command reads the value of the "Questionable Bandwidth Limit Channel Status Event Register" of the specified channel.

Variable

Parameter	*Value*
Description	The value of the Questionable Bandwidth Limit Channel Status Event Register
Data type	Long integer type (Long)

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).EVENt
```

Related objects

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).NTRansition

Description

This command sets/gets the value of the negative transition filter of the "Questionable Bandwidth Limit Channel Status Register" for the specified channel.

Variable

Parameter	*Value*
Description	The value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0, and bit 5 to 15 can not be set to 1.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).NTRansition = 16
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).NTRansition
```

Related objects

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}:NTRansition <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}:NTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:NTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.BLIMit.CHANnel(*Ch*).PTRansition

Description

This command sets/gets the value of the positive transition filter of the "Questionable Bandwidth Limit Channel Status Register" for the specified channel.

Variable

Parameter	*Value*
Description	The value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).PTRansition = 0
Stat = SCPI.STATus.QUEStionable.BLIMit.CHANnel(1).PTRansition
```

Related objects

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.BLIMit.CHANnel(Ch).NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}:PTRansition <numeric>

:STATus:QUEStionable:BLIMit:CHANnel{[1]-4}:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:PTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:CHAN1:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.BLIMit.CONDition

Description

This command reads the value of the Questionable Bandwidth Limit Status Condition Register.

Variable

Parameter	*Value*
Description	The value of the Questionable Bandwidth Limit Status Condition Register.
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.CONDition
```

Related objects

SCPI.STATus.QUEStionable.BLIMit.NTRansition

SCPI.STATus.QUEStionable.BLIMit.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.BLIMit.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.BLIMit.ENABle

Description

This command sets/gets the value of the Questionable Bandwidth Limit Status Enable Register.

Variable

Parameter	*Value*
Description	The value of the Questionable Bandwidth Limit Status Enable Register.
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.ENABle = 16
Stat = SCPI.STATus.QUEStionable.BLIMit.ENABle
```

Related objects

SCPI.STATus.QUEStionable.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:ENABle <numeric>

:STATus:QUEStionable:BLIMit:ENABle?

Query response

1974

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:ENAB 16"
20 OUTPUT 717;":STAT:QUES:BLIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.BLIMit.EVENt

Description

This command reads the value of the Questionable Bandwidth Limit Status Event Register.

Variable

Parameter	*Value*
Description	The value of the Questionable Bandwidth Limit Status Event Register.
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.BLIMit.EVENt
```

Related objects

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.BLIMit.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.BLIMit.NTRansition

Description

This command sets/gets the value of the negative transition filter of the Questionable Bandwidth Limit Status Register.

Variable

Parameter	*Value*
Description	The value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.BLIMit.NTRansition
```

Related objects

SCPI.STATus.QUEStionable.BLIMit.EVENt

SCPI.STATus.QUEStionable.BLIMit.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:NTRansition <numeric>

:STATus:QUEStionable:BLIMit:NTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:NTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.BLIMit.PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.BLIMit.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.BLIMit.PTRansition

Description

This command sets/gets the value of the positive transition filter of the Questionable Bandwidth Limit Status Register.

Variable

Parameter	*Value*
Description	The value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.BLIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.BLIMit.PTRansition
```

Related objects

SCPI.STATus.QUEStionable.BLIMit.EVENt

SCPI.STATus.QUEStionable.BLIMit.NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:BLIMit:PTRansition <numeric>

:STATus:QUEStionable:BLIMit:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:BLIM:PTR 16"
20 OUTPUT 717;":STAT:QUES:BLIM:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.CONDition

Description

This command reads the value of the Questionable Status Condition Register.

Variable

Parameter	*Value*
Description	Value of the Questionable Status Condition Register
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.CONDition
```

Related objects

SCPI.STATus.QUEStionable.NTRansition

SCPI.STATus.QUEStionable.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.ENABle

Description

This command sets/gets the value of the Questionable Status Enable Register.

Variable

Parameter	*Value*
Description	Value of the Questionable Status Enable Register
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0 to 7 and bit 12 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.ENABle = 1024
Stat = SCPI.STATus.QUEStionable.ENABle
```

Related objects

SCPI.IEEE4882.SRE

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:ENABle <numeric>

:STATus:QUEStionable:ENABle?

Query response

1982

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:ENAB 1024"
20 OUTPUT 717;":STAT:QUES:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.EVENt

Description

This command reads the value of the Questionable Status Event Register.

Variable

Parameter	*Value*
Description	Value of the Questionable Status Event Register
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.EVENt
```

Related objects

SCPI.IEEE4882.CLS

SCPI.STATus.QUEStionable.NTRansition

SCPI.STATus.QUEStionable.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).CONDition

Description

This command reads the value of the Questionable Limit Channel Status Condition Register of the specified channel (*Ch*).

Variable

Parameter	*Value*
Description	Value of the Questionable Limit Channel Status Condition Register
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).CONDition
```

Related objects

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).NTRansition

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ENABle = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).ENABle

Description

This command sets/gets the value of the Questionable Limit Channel Status Enable Register of the specified channel (*Ch*).

Variable

Parameter	*Value*
Description	Value of the Questionable Limit Channel Status Enable Register
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the channel/trace number.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ENABle = 16
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).ENABle
```

Related objects

SCPI.STATus.QUEStionable.LIMit.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}:ENABle <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}:ENABle?

Query response

1986

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ENAB 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).EVENt

Description

This command reads the value of the Questionable Limit Channel Status Event Register of the specified channel (*Ch*).

Variable

Parameter	*Value*
Description	Value of the Questionable Limit Channel Status Event Register of the specified channel
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).EVENt
```

Related objects

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).NTRansition

Description

This command sets/gets the value of the negative transition filter of the Questionable Limit Channel Status Register of the specified channel (*Ch*).

Variable

Parameter	*Value*
Description	Value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).NTRansition = 16
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).NTRansition
```

Related objects

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}:NTRansition <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}:NTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:NTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.CHANnel(*Ch*).PTRansition

Description

This command sets/gets the value of the positive transition filter of the Questionable Limit Channel Status Register of the specified channel (*Ch*).

Variable

Parameter	*Value*
Description	Value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the channel/trace number.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.CHANnel(1).PTRansition = 0
Stat = SCPI.STATus.QUEStionable.LIMit.CHANnel(1).PTRansition
```

Related objects

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.LIMit.CHANnel(Ch).NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}:PTRansition <numeric>

:STATus:QUEStionable:LIMit:CHANnel{[1]-4}:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:CHAN1:PTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:CHAN1:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.LIMit.CONDition

Description

This command reads the value of the Questionable Limit Status Condition Register.

Variable

Parameter	*Value*
Description	Value of the Questionable Limit Status Condition Register
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.CONDition
```

Related objects

SCPI.STATus.QUEStionable.LIMit.NTRansition

SCPI.STATus.QUEStionable.LIMit.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.LIMit.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.ENABle

Description

This command sets/gets the value of the Questionable Limit Status Enable Register.

Variable

Parameter	*Value*
Description	Value of the Questionable Limit Status Enable Register
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting for the channel/trace number.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.ENABle = 16
Stat = SCPI.STATus.QUEStionable.LIMit.ENABle
```

Related objects

SCPI.STATus.QUEStionable.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:ENABle <numeric>

:STATus:QUEStionable:LIMit:ENABle?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:ENAB 16"
20 OUTPUT 717;":STAT:QUES:LIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.LIMit.EVENt

Description

This command reads the value of the Questionable Limit Status Event Register.

Variable

Parameter	*Value*
Description	Value of the Questionable Limit Status Event Register
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.LIMit.EVENt
```

Related objects

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.LIMit.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.NTRansition

Description

This command sets/gets the value of the negative transition filter of the Questionable Limit Status Register.

Variable

Parameter	*Value*
Description	Value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.LIMit.NTRansition
```

Related objects

SCPI.STATus.QUEStionable.LIMit.EVENt

SCPI.STATus.QUEStionable.LIMit.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:NTRansition <numeric>

:STATus:QUEStionable:LIMit:NTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:NTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.LIMit.PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.LIMit.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.LIMit.PTRansition

Description

This command sets/gets the value of positive transition filter of the Questionable Limit Status Register.

Variable

Parameter	*Value*
Description	Value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the channel/trace number.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.LIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.LIMit.PTRansition
```

Related objects

SCPI.STATus.QUEStionable.LIMit.EVENt

SCPI.STATus.QUEStionable.LIMit.NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:LIMit:PTRansition <numeric>

:STATus:QUEStionable:LIMit:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:LIM:PTR 16"
20 OUTPUT 717;":STAT:QUES:LIM:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.NTRansition

Description

This command sets/gets the value of negative transition filter of the Questionable Status Register.

Variable

Parameter	*Value*
Description	Value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit (0 to 7) and bit (11 to 15) can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.NTRansition = 512
Stat = SCPI.STATus.QUEStionable.NTRansition
```

Related objects

SCPI.STATus.QUEStionable.EVENt

SCPI.STATus.QUEStionable.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:NTRansition <numeric>

:STATus:QUEStionable:NTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:NTR 512"
20 OUTPUT 717;":STAT:QUES:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.PTRansition

Description

This command sets/gets the value of the positive transition filter of the Questionable Status Register.

Variable

Parameter	*Value*
Description	Value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	1024
Note	The bit 0 to 7 and bit 11 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.PTRansition = 512
Stat = SCPI.STATus.QUEStionable.PTRansition
```

Related objects

SCPI.STATus.QUEStionable.EVENt

SCPI.STATus.QUEStionable.NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:PTRansition <numeric>

:STATus:QUEStionable:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:PTR 512"
20 OUTPUT 717;":STAT:QUES:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).CONDition

Description

This command reads the value of the Questionable Ripple Limit Channel Status Condition Register for the specified channel.

Variable

Parameter	*Value*
Description	The value of the Questionable Ripple Limit Channel Status Condition Register.
Data type	Long integer type (Long)

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).CONDition
```

Related objects

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).NTRansition

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).ENABle = *Value*

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).ENABle

Description

This command sets/gets the value of the Questionable Ripple Limit Channel Status Enable Register for the specified channel.

Variable

Parameter	*Value*
Description	The value of the Questionable Ripple Limit Channel Status Enable Register.
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ENABle = 16
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).ENABle
```

Related objects

SCPI.STATus.QUEStionable.RLIMit.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}:ENABle <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}:ENABle?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ENAB 16"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).EVENt

Description

This command reads the value of the Questionable Ripple Limit Channel Status Event Register for the specified channel.

Variable

Parameter	*Value*
Description	The value of the Questionable Ripple Limit Channel Status Event Register.
Data type	Long integer type (Long)

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).EVENt
```

Related objects

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).NTRansition

Description

This command sets/gets the value of the negative transition filter of the Questionable Ripple Limit Channel Status Register for the specified channel.

Variable

Parameter	*Value*
Description	The value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0, and bit 5 to 15 can not be set to 1.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).NTRansition = 16
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).NTRansition
```

Related objects

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}:NTRansition <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}:NTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:NTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.RLIMit.CHANnel(*Ch*).PTRansition

Description

This command sets/gets the value of the positive transition filter of the Questionable Ripple Limit Channel Status Register for the specified channel.

Variable

Parameter	*Value*
Description	The value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

For information on the variable (*Ch*), see Ch.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).PTRansition = 0
Stat = SCPI.STATus.QUEStionable.RLIMit.CHANnel(1).PTRansition
```

Related objects

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).EVENt

SCPI.STATus.QUEStionable.RLIMit.CHANnel(Ch).NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}:PTRansition <numeric>

:STATus:QUEStionable:RLIMit:CHANnel{[1]-4}:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:PTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:CHAN1:PTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.CONDition

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.RLIMit.CONDition

Description

This command reads the value of the Questionable Ripple Limit Status Condition Register.

Variable

Parameter	*Value*
Description	The value of the Questionable Ripple Limit Status Condition Register.
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat = SCPI.STATus.QUEStionable.RLIMit.CONDition
```

Related objects

SCPI.STATus.QUEStionable.RLIMit.NTRansition

SCPI.STATus.QUEStionable.RLIMit.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:CONDition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:COND?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.ENABle

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.RLIMit.ENABle = *Value*

Value = SCPI.STATus.QUEStionable.RLIMit.ENABle

Description

This command sets/gets the value of the Questionable Ripple Limit Status Enable Register.

Variable

Parameter	*Value*
Description	The value of the Questionable Ripple Limit Status Enable Register.
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.ENABle = 16
Stat = SCPI.STATus.QUEStionable.RLIMit.ENABle
```

Related objects

SCPI.STATus.QUEStionable.ENABle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:ENABle <numeric>

:STATus:QUEStionable:RLIMit:ENABle?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:ENAB 16"
20 OUTPUT 717;":STAT:QUES:RLIM:ENAB?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.EVENt

Object type

Property (**Read Only**)

Syntax

Value = SCPI.STATus.QUEStionable.RLIMit.EVENt

Description

This command reads the value of the Questionable Ripple Limit Status Event Register.

Variable

Parameter	*Value*
Description	The value of the Questionable Ripple Limit Status Event Register.
Data type	Long integer type (Long)

Examples

```
Dim Stat As Long
Stat =SCPI.STATus.QUEStionable.RLIMit.EVENt
```

Related objects

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit[:EVENt]?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM?"
20 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.NTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.RLIMit.NTRansition = *Value*

Value = SCPI.STATus.QUEStionable.RLIMit.NTRansition

Description

This command sets/gets the value of the negative transition filter of the Questionable Ripple Limit Status Register.

Variable

Parameter	*Value*
Description	The value of the negative transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	0
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.NTRansition = 6
Stat = SCPI.STATus.QUEStionable.RLIMit.NTRansition
```

Related objects

SCPI.STATus.QUEStionable.RLIMit.EVENt

SCPI.STATus.QUEStionable.RLIMit.PTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:NTRansition <numeric>

:STATus:QUEStionable:RLIMit:NTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:NTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:NTR?"
30 ENTER 717;A
```

SCPI.STATus.QUEStionable.RLIMit.PTRansition

Object type

Property (**Read-Write**)

Syntax

SCPI.STATus.QUEStionable.RLIMit.PTRansition = *Value*

Value = SCPI.STATus.QUEStionable.RLIMit.PTRansition

Description

This command sets/gets the value of the positive transition filter of the Questionable Ripple Limit Status Register.

Variable

Parameter	*Value*
Description	The value of the positive transition filter
Data type	Long integer type (Long)
Range	0 to 65535
Preset value	Varies depending on the upper limit setting of the number of channels/traces.
Note	The bit 0, and bit 5 to 15 can not be set to 1.

Examples

```
Dim Stat As Long
SCPI.STATus.QUEStionable.RLIMit.PTRansition = 6
Stat = SCPI.STATus.QUEStionable.RLIMit.PTRansition
```

Related objects

SCPI.STATus.QUEStionable.RLIMit.EVENt

SCPI.STATus.QUEStionable.RLIMit.NTRansition

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:STATus:QUEStionable:RLIMit:PTRansition <numeric>

:STATus:QUEStionable:RLIMit:PTRansition?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":STAT:QUES:RLIM:PTR 16"
20 OUTPUT 717;":STAT:QUES:RLIM:PTR?"
30 ENTER 717;A
```

SYSTEM

SCPI.SYSTem.BACKlight

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.BACKlight = *Status*

Status = SCPI.SYSTem.BACKlight

Description

This command turns ON/OFF or return the status of the backlight of the LCD display.

NOTE When the backlight is OFF, you cannot read the information on the display.

Variable

Parameter	*Status*
Description	ON/OFF of the backlight
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the backlight. • False or OFF: Turns OFF the backlight.
Preset value	True or ON

Examples

```
Dim BckLght As Boolean
SCPI.SYSTem.BACKlight = False
BckLght = SCPI.SYSTem.BACKlight
```

Equivalent key

System > **Backlight**

NOTE To turn the backlight ON, press any key on the front panel.

Equivalent SCPI command

Syntax

:SYSTem:BACKlight {ON|OFF|1|0}

:SYSTem:BACKlight?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:BACK OFF"
20 OUTPUT 717;":SYST:BACK?"
30 ENTER 717;A
```

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

Object type

Method (**Write Only**)

Syntax

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

Description

This command generates a beep for the notification of the completion of an operation.

Examples

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

Related objects

SCPI.SYSTem.BEEPer.COMPlete.STATe

SCPI.SYSTem.BEEPer.WARNing.IMMediate

Equivalent key

System > **Misc Setup** > **Beeper** > **Test Beep Complete**

Equivalent SCPI command

Syntax

:SYSTem:BEEPer:COMPlete:IMMediate

Example of use

```
10 OUTPUT 717;":SYST:BEEP:COMP:IMM"
```

SCPI.SYSTem.BEEPer.COMPlete.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.BEEPer.COMPlete.STATe = *Status*

Status = SCPI.SYSTem.BEEPer.COMPlete.STATe

Description

This command turns ON/OFF or returns the status of the beeper for the notification of the completion of the operation.

Variable

Parameter	*Status*
Description	ON/OFF of the beeper for the notification of the completion of the operation
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the beeper for the notification of the completion of the operation. • False or OFF: Turns OFF the beeper for the notification of the completion of the operation.
Preset value	True or ON

Examples

```
Dim BeepComp As Boolean
SCPI.SYSTem.BEEPer.COMPlete.STATe = False
BeepComp = SCPI.SYSTem.BEEPer.COMPlete.STATe
```

Related objects

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

SCPI.SYSTem.BEEPer.WARNing.STATe

Equivalent key

System > **Misc Setup** > **Beeper** > **Beep Complete**

Equivalent SCPI command

Syntax

:SYSTem:BEEPer:COMPlete:STATe {ON|OFF|1|0}

:SYSTem:BEEPer:COMPlete:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:BEEP:COMP:STAT OFF"
20 OUTPUT 717;":SYST:BEEP:COMP:STAT?"
30 ENTER 717;A
```

SCPI.SYSTem.BEEPer.WARNing.IMMediate

Object type

Method (**Write Only**)

Syntax

SCPI.SYSTem.BEEPer.WARNing.IMMediate

Description

This command generates a beep for the notification of warning/limit test results.

Examples

SCPI.SYSTem.BEEPer.WARNing.IMMediate

Related objects

SCPI.SYSTem.BEEPer.WARNing.STATe

SCPI.SYSTem.BEEPer.COMPlete.IMMediate

Equivalent key

System > **Misc Setup** > **Beeper** > **Test Beep Warning**

Equivalent SCPI command

Syntax

:SYSTem:BEEPer:WARNing:IMMediate

Example of use

```
10 OUTPUT 717;":SYST:BEEP:WARN:IMM"
```

SCPI.SYSTem.BEEPer.WARNing.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.BEEPer.WARNing.STATe = *Status*

Status = SCPI.SYSTem.BEEPer.WARNing.STATe

Description

This command turns ON/OFF or return the status of the beeper for the notification of warning/limit test results.

Variable

Parameter	*Status*
Description	ON/OFF of the beeper for the notification of warning/limit test result
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the beeper for the notification of warning/limit test result. • False or OFF: Turns OFF the beeper for the notification of warning/limit test result.
Preset value	True or ON

Examples

```
Dim BeepWarn As Boolean
SCPI.SYSTem.BEEPer.WARNing.STATe = False
BeepWarn = SCPI.SYSTem.BEEPer.WARNing.STATe
```

Related objects

SCPI.SYSTem.BEEPer.WARNing.IMMediate

SCPI.SYSTem.BEEPer.COMPlete.STATe

Equivalent key

System > **Misc Setup** > **Beeper** > **Beep Warning**

Equivalent SCPI command

Syntax

:SYSTem:BEEPer:WARNing:STATe {ON|OFF|1|0}

:SYSTem:BEEPer:WARNing:STATe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:BEEP:WARN:STAT OFF"
20 OUTPUT 717;":SYST:BEEP:WARN:STAT?"
30 ENTER 717;A
```

SCPI.SYSTem.COMMunicate.LAN.CONTrol

Object Type

Property (**Read Only**)

Syntax

Value = SCPI.SYSTem.COMMunicate.LAN.CONTrol

Description

This command gets control port number of socket for LAN connection.

Variable

Parameter	*Value*
Description	control port number of socket
Data Type	Long integer type (Long)

Examples

```
Dim PortNum as Long
PortNum = SCPI.SYSTem.COMMunicate.LAN.CONTrol
```

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SYSTem:COMMunicate:LAN:CONTrol?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:COMM:LAN:CONT?"
20 ENTER 717;A
```

SCPI.SYSTem.DATE

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.DATE = *Data*

Data = SCPI.SYSTem.DATE

Description

This command sets/gets the date of the clock built in the E5061B.

Variable

Parameter	*Data*
Description	Indicates 3-element array data (date of the built-in clock). • *Data(0)* Sets year. • *Data(1)* Sets month. • *Data(2)* Sets day. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	• *Data(0)* 1980 to 2099 • *Data(1)* 1 to 12 • *Data(2)* 1 to 31
Resolution	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Day As Variant
SCPI.SYSTem.DATE = Array(2001,12,24)
Day = SCPI.SYSTem.DATE
```

```
Dim Day(2) As Variant
Dim Ref As Variant
Day(0) = 2001
Day(1) = 12
Day(2) = 24
SCPI.SYSTem.DATE = Day
Ref = SCPI.SYSTem.DATE
```

Related objects

SCPI.SYSTem.TIME

SCPI.DISPlay.CLOCk

Equivalent key

System > **Misc Setup** > **Clock Setup** > **Set Date and Time**

Equivalent SCPI command

Syntax

```
:SYSTem:DATE <numeric 1>,<numeric 2>,<numeric 3>
:SYSTem:DATE?
```

Query response

```
{numeric 1},{numeric 2},{numeric 3}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SYST:DATE 2002,1,1"
20 OUTPUT 717;":SYST:DATE?"
30 ENTER 717;A,B,C
```

SCPI.SYSTem.ERRor

Object type

Property (**Read Only**)

Syntax

Err = SCPI.SYSTem.ERRor

Description

This command reads the oldest error from the list of errors stored in the error queue of the E5061B. The read-out error is deleted from the error queue. The size of the error queue is 100.

Executing SCPI.IEEE4882.CLS command clears the errors stored in the error queue.

NOTE This object can not return an error that occurs by the manual operation or the SCPI command used in controlling the E5061B from the external con-troller.

Variable

Parameter	*Err*
Description	Indicates 2-element array data (for error). • *Err(0)* :Error number • *Err(1)* :Error message The index of the array starts from 0.
Data type	Variant type (Variant)
Note	If no error is stored in the error queue, 0 and "No error" are read out as the error number and the error message.

Examples

```
Dim Err As Variant
Err = SCPI.SYSTem.ERRor
```

Related objects

SCPI.IEEE4882.CLS

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:SYSTem:ERRor?

Query response

{numeric},{string}<newline><^END>

{numeric}:

Error number

{string}:

Error message (a character string with double quotation marks (""))

If no error is stored in the error queue, 0 and "No error" are read out as the error number and the error message, respectively.

Example of use

```
10 OUTPUT 717;":SYST:ERR?"
20 ENTER 717;A,B$
```

SCPI.SYSTem.ISPControl.DCBias.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.ISPControl.DCBias.STATe = *Status*

Status = SCPI.SYSTem.ISPControl.DCBias.STATe

Description

This command turns ON/OFF or returns the status of the Initial Source Port Control feature for DC Bias (to switch the stimulus output in the trigger hold state to a test port).

Variable

Parameter	*Status*
Description	ON/OFF Initial Source Port Control feature for DC Bias
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: ON Control feature. • False or OFF: OFF Control feature.
Preset value	True or ON

Examples

```
SCPI.INITiate(1).CONTinuous = False
SCPI.SYSTem.ISPControl.DCBias.STATe = True
```

Related objects

SCPI.SYSTem.ISPControl.STAT

Equivalent key

System > **Service Menu** > **Init Src Ctrl** > **DC Bias**

Equivalent SCPI command

Syntax

:SYSTem:ISPControl:DCBias:STATe {ON|OFF|1|0}

:SYSTem:ISPControl:DCBias:STATe ?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":INIT:CONT OFF"
20 OUTPUT 717;":SYST:ISPC:DCBI:STAT ON"
```

SCPI.SYSTem.ISPControl.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.ISPControl.STATe = *Status*

Status = SCPI.SYSTem.ISPControl.STATe

Description

This command turns ON/OFF or returns the status of the Initial Source Port Control feature for RF signal (to switch the stimulus output in the trigger hold state to a test port).

Variable

Parameter	*Status*
Description	ON/OFF Initial Source Port Control feature for RF signal
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: ON Control feature. • False or OFF: OFF Control feature.
Preset value	True or ON

Examples

```
SCPI.INITiate(1).CONTinuous = False
SCPI.SYSTem.ISPControl.STATe = True
```

Related objects

SCPI.SYSTem.ISPControl.DCBias.STATe

Equivalent key

System > **Service Menu** > **Init Src Ctrl** > **RF Out**

Equivalent SCPI command

Syntax

:SYSTem:ISPControl[:STATe] {ON|OFF|1|0}

:SYSTem:ISPControl[:STATe] ?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":INIT:CONT OFF"
20 OUTPUT 717;":SYST:ISPC ON"
```

SCPI.SYSTem.KLOCk.KBD

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.KLOCk.KBD = *Status*

Status = SCPI.SYSTem.KLOCk.KBD

Description

This command sets/gets whether to lock the operation of the front panel (key and rotary knob) and keyboard.

Variable

Parameter	*Status*
Description	ON/OFF of lock
Data type	Boolean type (Boolean)
Range	Select from either of the following. • True or ON: Specifies lock. • False or OFF: Specifies unlock.
Preset value	False or OFF

Examples

```
Dim FKLock As Boolean
SCPI.SYSTem.KLOCk.KBD = True
FKLock = SCPI.SYSTem.KLOCk.KBD
```

Related objects

SCPI.SYSTem.KLOCk.MOUSe

Equivalent key

System > **Misc Setup** > **Key Lock** > **Front Panel & Keyboard Lock**

Equivalent SCPI command

Syntax

:SYSTem:KLOCk:KBD {ON|OFF|1|0}

:SYSTem:KLOCk:KBD?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:KLOC:KBD ON"
20 OUTPUT 717;":SYST:KLOC:KBD?"
30 ENTER 717;A
```

SCPI.SYSTem.KLOCk.MOUSe

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.KLOCk.MOUSe = *Status*

Status = SCPI.SYSTem.KLOCk.MOUSe

Description

This command sets/gets whether to lock the operation of the mouse and touch screen.

Variable

Parameter	*Status*
Description	ON/OFF of lock
Data type	Boolean type (Boolean)
Range	Select from either of the following. • True or ON: Specifies lock. • False or OFF: Specifies unlock.
Preset value	False or OFF

Examples

```
Dim MTLock As Boolean
SCPI.SYSTem.KLOCk.MOUSe = True
MTLock = SCPI.SYSTem.KLOCk.MOUSe
```

Related objects

SCPI.SYSTem.KLOCk.KBD

Equivalent key

System > **Misc Setup** > **Key Lock** > **Touch Screen & Mouse Lock**

Equivalent SCPI command

Syntax

:SYSTem:KLOCk:MOUSe {ON|OFF|1|0}

:SYSTem:KLOCk:MOUSe?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:KLOC:MOUS ON"
20 OUTPUT 717;":SYST:KLOC:MOUS?"
30 ENTER 717;A
```

SCPI.SYSTem.POFF

Object type

Method (**Write Only**)

Syntax

SCPI.SYSTem.POFF

Description

This command turns OFF the E5061B.

Examples

SCPI.SYSTem.POFF

Equivalent key

Standby switch

Equivalent SCPI command

Syntax

:SYSTem:POFF

Example of use

```
10 OUTPUT 717;":SYST:POFF"
```

SCPI.SYSTem.PRESet

Object type

Method (**Write Only**)

Syntax

SCPI.SYSTem.PRESet

Description

This command presets the setting state of the E5061B to the original factory setting (Default Conditions). This command is different from SCPI.IEEE4882.RSTas the continuous startup mode (see SCPI.INITiate(Ch).CONTinuous) of channel 1 is set to ON.

Examples

SCPI.SYSTem.PRESet

Related objects

SCPI.IEEE4882.RST

Equivalent key

Preset > **OK**

Equivalent SCPI command

Syntax

:SYSTem:PRESet

Example of use

```
10 OUTPUT 717;":SYST:PRES"
```

SCPI.SYSTem.SECurity.LEVel

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.SECurity.LEVel = *Param*

Param = SCPI.SYSTem.SECurity.LEVel

Description

This command sets/gets the security level.

Variable

Parameter	*Param*
Description	Security level.
Data type	Character string type (String)
Range	Select from either of the following: • "NONE": Specifies OFF to the security level. • "LOW": Specifies LOW level to the security level. • "HIGH": Specifies HIGH level to the security level.
Preset value	"NONE"
Note	When the setting is LOW, it is able to change to NONE or HIGH. But when this setting is HIGH, it is not able to change NONE or LOW. The setting can be turned NONE by executing the preset or recalling when the setting of frequency blank function is HIGH. Even if the setting is LOW and HIGH, the command that reads out the frequency is not influenced.

Examples

```
Dim SecLev As String
SCPI.SYSTem.SECurity.LEVel = "LOW"
SecLev = SCPI.SYSTem.SECurity.LEVel
```

Equivalent key

System > **Service Menu** > **Security Level** > **None**|**Low**|**High**

Equivalent SCPI command

Syntax

```
:SYSTem:SECurity:LEVel {NONE|LOW|HIGH}
:SYSTem:SECurity:LEVel?
```

Query response

```
{NONE|LOW|HIGH}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":SYST:SEC:LEV LOW"
20 OUTPUT 717;":SYST:SEC:LEV?"
30 ENTER 717;A$
```

SCPI.SYSTem.SERVice

Object type

Property (**Read Only**)

Syntax

Status = SCPI.SYSTem.SERVice

Description

This command reads whether the E5061B is in the service mode or not.

Variable

Parameter	*Status*
Description	Whether to be in the service mode
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: In the service mode. • False or OFF : Not in the service mode.
Preset Value	False or OFF

Examples

```
Dim SvMode As Boolean
SvMode = SCPI.SYSTem.SERVice
```

Equivalent key

Displayed on the instrument status bar (at the bottom of the LCD display).

Equivalent SCPI command

Syntax

:SYSTem:SERVice?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:SERV?"
30 ENTER 717;A
```

SCPI.SYSTem.SET

Object Type

Method (**Write Only**)

Syntax

SCPI.SYSTem.SET = *Value*

Description

This command recalls the state of the instrument when the *LRN? query is executed. The contents to be recalled depends on the contents of the block data.

If the block data contains trace state, the trigger source (:SCPI.TRIGger.SEQuence.SOURce) becomes "MANual". The result of *LRN? query contains ":SYSTem:SET " prefix. Hence, the *LRN? simply executes this command.

This command requires instrument settings by binary block data (same as Save/Recall state file contents).

Variable

Parameter	*Value*
Description	This command recalls the state of the instrument when the *LRN? query is executed.
Data Type	Binary (byte)

Related objects

SCPI.IEEE4882.LRN

Equivalent Key

No equivalent key is available on the front panel.

Equivalent SCPI Command

Syntax

:SYSTem:SET

Example of use (VISA-COM)

```
Dim SETData() As Byte, NoofByte As Double

'*** Recall the State data from the file, State01.sta is a state file saved by E5061B
Open "C:\State01.sta" For Binary As #1
```

```
    NoofByte = LOF(1)
    ReDim SETData(NoofByte)
    Get #1, , SETData()
Close

'*** Send the State file data to E5061B
Age506x.WriteIEEEBlock ":SYST:SET ", SETData, True
```

SCPI.SYSTem.TIME

Object type

Property (**Read-Write**)

Syntax

SCPI.SYSTem.TIME = *Data*

Data = SCPI.SYSTem.TIME

Description

This command sets/gets the time of the clock built in the E5061B.

Variable

Parameter	*Data*
Description	Indicates 3-element array data (time of the built-in clock). *Data(0):* Sets hour (24-hour basis) *Data(1)* :Sets minute. *Data(2)* :Sets second. The index of the array starts from 0.
Data type	Variant type (Variant)
Range	*Data(0)* :0 to 23 *Data(1)* :0 to 59 *Data(2)* :0 to 59
Resolution	1
Note	If the specified variable is out of the allowable setup range, the minimum value (if the lower limit of the range is not reached) or the maximum value (if the upper limit of the range is exceeded) is set.

Examples

```
Dim Time As Variant
SCPI.SYSTem.TIME = Array(21,30,0)
Time = SCPI.SYSTem.TIME
```

```
Dim Time(2) As Variant
Dim Ref As Variant
Time(0) = 21
Time(1) = 30
Time(2) = 0
SCPI.SYSTem.TIME = Time
Ref = SCPI.SYSTem.TIME
```

Related objects

SCPI.SYSTem.DATE

SCPI.DISPlay.CLOCk

Equivalent key

System > **Misc Setup** > **Clock Setup** > **Set Date and Time**

Equivalent SCPI command

Syntax

:SYSTem:TIME <numeric 1>,<numeric 2>,<numeric 3>

:SYSTem:TIME?

Query response

{numeric 1},{numeric 2},{numeric 3}<newline><^END>

Example of use

```
10 OUTPUT 717;":SYST:TIME 17,30,0"
20 OUTPUT 717;":SYST:TIME?"
30 ENTER 717;A,B,C
```

SCPI.SYSTem.UPReset

Object type

Method (**Write Only**)

Syntax

SCPI.SYSTem.UPReset

Description

This command presets the E5061B with the user settings. The command is executed regardless of the operation mode in preset state.

NOTE If you try to specify a file for a preset (D:\UserPreset.sta) that does not exist, a warning message is displayed and SCPI.SYSTem.PRESet is executed.

Examples

SCPI.SYSTem.UPReset

Related objects

SCPI.IEEE4882.RST

SCPI.SYSTem.PRESet

Equivalent key

Preset > **OK**

Equivalent SCPI command

Syntax

:SYSTem:UPReset

Example of use

```
10 OUTPUT 717;":SYST:UPR"
```

TRIGGER

SCPI.TRIGger.OUTPut.DURation

Object Type

Property (**Read-Write**)

Syntax

SCPI.TRIGger.OUTPut.DURation = *Value*

Value = SCPI.TRIGger.OUTPut.DURation

Description

This command sets/gets pulse width of output trigger signal.

Variable

Parameter	*Value*
Description	pulse width
Data Type	Double precision floating point type (Double)
Range	1µ to 1
Preset Value	1µ
Unit	sec
Resolution	1µ

Examples

```
Dim WidTrig as Double
SCPI.TRIGger.OUTPut.DURation = 2E-6
WidTrig = SCPI.TRIGger.OUTPut.DURation
```

Related Objects

Equivalent Key

Trigger > **Pulse Width**

Equivalent SCPI Command

Syntax

:TRIGger:OUTPut:DURation <numeric>

:TRIGger:OUTPut:DURation?

Query Response

{numeric} <newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:OUTP:DUR 3E-6"
20 OUTPUT 717;":TRIG:OUTP:DUR?"
30 ENTER 717;A
```

SCPI.TRIGger.OUTPut.POLarity

Object type

Property (**Read-Write**)

Syntax

SCPI.TRIGger.OUTPut.POLarity=*Param*

Param=SCPI.TRIGger.OUTPut.POLarity

Description

This command sets/gets the polarity of the pulse generated from the External Trigger (Output).

Variable

Parameter	*Param*
Description	Sets/Gets polarity of the pulse generated by the output trigger.
Data type	Character string type (String)
Range	Select from either of the following: • "POSitive": Generates a Positive pulse. • "NEGative": Generates a Negative pulse.
Preset value	"POSitive"

Examples

```
Dim TrigPol As String
TrigPol="NEGative"
SCPI.TRIGger.OUTPut.POLarity=TrigPol
```

Related objects

SCPI.TRIGger.OUTPut.POSition

SCPI.TRIGger.OUTPut.STATe

Equivalent key

Trigger > **Polarity**

Equivalent SCPI command

Syntax

:TRIGger:OUTPut:POLarity {POSitive|NEGative}

:TRIGger:OUTPut:POLarity?

Query response

{POS|NEG}<newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:OUTP:POL POS"
20 OUTPUT 717;":TRIG:OUTP:POL?"
30 ENTER 717;A$
```

SCPI.TRIGger.OUTPut.POSition

Object type

Property (**Read-Write**)

Syntax

SCPI.TRIGger.OUTPut.POSition=*Param*

Param=SCPI.TRIGger.OUTPut.POSition

Description

This command sets/gets the position of the External Trigger Output Port.

Variable

Parameter	*Param*
Description	Sets/Gets (after or before measurement point) position of the output trigger.
Data type	Character string type (String)
Range	Select from either of the following: • "AFTer": Generates a Pulse (trigger) after the measurement points. • "BEFore": Generates a Pulse (trigger) before the measurement points.
Preset value	"AFTer"

Examples

```
Dim TrigPos As String
TrigPos="BEFore"
SCPI.TRIGger.OUTPut.POSition=TrigPos
```

Related objects

SCPI.TRIGger.OUTPut.POLarity

SCPI.TRIGger.OUTPut.STATe

Equivalent key

Trigger > **Position**

Equivalent SCPI command

Syntax

```
:TRIGger:OUTPut:POSition {AFTer|BEFore}
:TRIGger:OUTPut:POSition?
```

Query response

```
{AFT|BEF}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":TRIG:OUTP:POS AFT"
20 OUTPUT 717;":TRIG:OUTP:POS?"
30 ENTER 717;A$
```

SCPI.TRIGger.OUTPut.STATe

Object type

Property (**Read-Write**)

Syntax

SCPI.TRIGger.OUTPut.STATe=*Status*

Status=SCPI.TRIGger.OUTPut.STATe

Description

This command sets/gets the External Trigger (Output) Port state.

Variable

Parameter	*Status*
Description	Sets/Gets external trigger output state.
Data type	Boolean type (Boolean)
Range	Select from either of the following: • True or ON: Turns ON the External Trigger Output • False or OFF : Turns OFF the External Trigger Output
Preset value	False or OFF

Examples

```
Dim TrigStat As boolean
TrigStat = True
SCPI.TRIGger.OUTPut.STATe=TrigStat
```

Related objects

SCPI.TRIGger.OUTPut.POLarity

SCPI.TRIGger.OUTPut.POSition

Equivalent key

Trigger > **Ext Trig Output**

Equivalent SCPI command

Syntax

:TRIGger:OUTPut[:STATe] {ON|1|OFF|0}

:TRIGger:OUTPut[:STATe]?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:OUTP ON"
20 OUTPUT 717;":TRIG:OUTP?"
30 ENTER 717;A$
```

SCPI.TRIGger.SEQuence.AVERage

Type of object

Property (**Read-Write**)

Syntax

SCPI.TRIGger.SEQuence.AVERage = *Status*

Status = SCPI.TRIGger.SEQuence.AVERage

Description

This command turns ON/OFF or gets the status of the averaging trigger function.

NOTE The sweep averaging feature must be set to *ON* when turning ON the averaging trigger feature.

Variable

Parameter	*Status*
Description	ON/OFF status of the averaging trigger
Data type	Boolean type (Boolean)
Range	Select from either of the following. • True or ON: Turns ON the averaging trigger. • False or OFF: Turns OFF the averaging trigger.
Preset value	False or OFF

Example of use

```
Dim Avetrig As Boolean
SCPI.TRIGger.SEQuence.AVERage = True
Avetrig = TRIGger.SEQuence.AVERage
```

Related objects

SCPI.SENSe(Ch).AVERage.STATe

Equivalent key

Ave > **Avg Trigger**

Equivalent SCPI command

Syntax

:TRIGger[:SEQuence]:AVERage {ON|OFF|1|0}

:TRIGger[:SEQuence]:AVERage?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:AVER ON"
20 OUTPUT 717;":TRIG:AVER?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.EXTernal.DELay

Type of object

Property (**Read-Write**)

Syntax

SCPI.TRIGger.SEQuence.EXTernal.DELay = *Value*

Value = SCPI.TRIGger.SEQuence.EXTernal.DELay

Description

This command sets/gets the time that it takes from receiving the trigger to starting the measurement when the trigger source is external.

Variable

Parameter	*Value*
Description	External trigger delay time
Data type	Double precision floating point type (Double)
Range	0 to 1
Preset value	0
Unit	s (second)
Resolution	1E-5

Example of use

```
Dim Extdel As Double
SCPI.TRIGger.SEQuence.EXTernal.DELay = 0.05
Extdel = SCPI.TRIGger.SEQuence.EXTernal.DELay
```

Related objects

SCPI.TRIGger.SEQuence.POINt

Equivalent key

Trigger > **Trigger Delay**

Equivalent SCPI command

Syntax

:TRIGger[:SEQuence]:EXTernal:DELay <numeric>

:TRIGger[:SEQuence]:EXTernal:DELay?

Query response

{numeric}<newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:EXT:DEL 0.05"
20 OUTPUT 717;":TRIG:EXT:DEL?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.EXTernal.SLOPe

Object type

Property (**Read-Write**)

Syntax

SCPI.TRIGger.SEQuence.EXTernal.SLOPe=*Param*

Param=*S*CPI.TRIGger.SEQuence.EXTernal.SLOPe

Description

This command sets/gets the polarity of the External Trigger (Input).

Variable

Parameter	*Param*
Description	Sets/Gets polarity of the external trigger (input).
Data type	Character string type (String)
Range	Select from either of the following: • "POSitive": Sets/Gets Positive polarity. • "NEGative": Sets/Gets Negative polarity.
Preset value	"NEGative"

Examples

```
Dim InputTrigPol As String
InputTrigPol="NEGative"
SCPI.TRIGger.SEQuence.EXTernal.SLOPe=InputTrigPol
```

Related objects

SCPI.TRIGger.SEQuence.EXTernal.DELay

Equivalent key

Trigger > **Ext Trig Input**

Equivalent SCPI command

Syntax

:TRIGger[:SEQuence]:EXTernal:SLOPe {POSitive|NEGative}

:TRIGger[:SEQuence]:EXTernal:SLOPe?

Query response

{POS|NEG}<newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:EXT:SLOP POS"
20 OUTPUT 717;":TRIG:EXT:SLOP?"
30 ENTER 717;A$
```

SCPI.TRIGger.SEQuence.IMMediate

Object type

Method (**Write Only**)

Syntax

SCPI.TRIGger.SEQuence.IMMediate

Description

This command generates a trigger immediately and executes a measurement, regardless of the setting of the trigger mode.

This command is different from SCPI.TRIGger.SEQuence.SINGle as the execution of the object finishes at the time of a trigger.

NOTE If you execute this object when the trigger system is not in the trigger wait state (trigger event detection state), an error occurs when executed and the object is ignored.

Examples

```
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.INITiate(1).CONTinuous = True
SCPI.TRIGger.SEQuence.IMMediate
```

Related objects

SCPI.TRIGger.SEQuence.SINGle

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:TRIGger[:SEQuence][:IMMediate]

Example of use

```
10 OUTPUT 717;":TRIG:IMM"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.POINt

Type of object

Property (**Read-Write**)

Syntax

SCPI.TRIGger.SEQuence.POINt = *Status*

Status = SCPI.TRIGger.SEQuence.POINt

Description

This command turns ON/OFF or returns the status of the point trigger feature.

NOTE When the trigger source is set to the internal trigger (Internal), the setting is ignored.

Variable

Parameter	*Status*
Description	Turns ON/OFF the point trigger
Data type	Boolean type (Boolean)
Range	Select from either of the following: True or ON: Turns ON the point trigger. False or OFF: Turns OFF the point trigger.
Preset value	False or OFF

Example of use

```
Dim Ptrig As Boolean
SCPI.TRIGger.SEQuence.POINt = True
Ptrig = TRIGger.SEQuence.POINt
```

Related objects

SCPI.TRIGger.SEQuence.SOURce

Equivalent key

Trigger > **Trigger Event**

Equivalent SCPI command

Syntax

:TRIGger[:SEQuence]:POINt {ON|OFF|1|0}

:TRIGger[:SEQuence]:POINt?

Query response

{1|0}<newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:POIN ON"
20 OUTPUT 717;":TRIG:POIN?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.SCOPe

Object type

Property (**Read-Write**)

Syntax

SCPI.TRIGger.SEQuence.SCOPe = *Param*

Param = SCPI.TRIGger.SEQuence.SCOPe

Description

This command sets/gets the effective scope of triggering. When this function is enabled with a value of "ACTive", only active channel is triggered. When this function is enabled with a value of "ALL", all channels of the E5061B are triggered.

For example, when this function is set at"ACTive" and SCPI.INITiate(Ch).CONTinuous is ON for all channels, a measurement channel is automatically changed by switching over the active channel.

Variable

Parameter	*Param*
Description	Trigger source
Data type	Character string type (String)
Range	Select from either of the following: • "ALL": Specifies trigger to all channels. • "ACTive": Specifies trigger to active channel.
Preset value	"ALL"

Examples

```
Dim TrigScope As Enum
SCPI.TRIGger.SEQuence.SCOPe = "ACTive"
TrigScope = SCPI.TRIGger.SEQuence.SCOPe
```

Related Objects

SCPI.INITiate(Ch).CONTinuous

Equivalent key

Trigger > **Trigger Scope**

Equivalent SCPI command

Syntax

:TRIGger[:SEQuence]:SCOPe {ALL|ACTive}

:TRIGger[:SEQuence]:SCOPe?

Query response

{ALL|ACT}<newline><^END>

Example of use

```
10 OUTPUT 717;":TRIG:SCOP ACT"
20 OUTPUT 717;":TRIG:SCOP?"
30 ENTER 717;A$
```

SCPI.TRIGger.SEQuence.SINGle

Object type

Method (**Write Only**)

Syntax

SCPI.TRIGger.SEQuence.SINGle

Description

This command generates a trigger immediately and executes a measurement, regardless of the setting of the trigger mode.

This command is different from SCPI.TRIGger.SEQuence.IMMediate as the execution of the object finishes when the measurement (all of the sweep) initiated with this object is complete. In other words, you can wait for the end of the measurement using the SCPI.IEEE4882.OPC object.

If you execute this object when the trigger system is not in the trigger wait state (trigger event detection state), an error occurs when executed and the object is ignored.

For details about the trigger system, see Trigger System.

Examples

```
Dim Dmy As Long
SCPI.TRIGger.SEQuence.SOURce = "bus"
SCPI.INITiate(1).CONTinuous = True
SCPI.TRIGger.SEQuence.SINGle
Dmy = SCPI.IEEE4882.OPC
```

Related objects

SCPI.TRIGger.SEQuence.IMMediate

SCPI.IEEE4882.OPC

Equivalent key

No equivalent key is available on the front panel.

Equivalent SCPI command

Syntax

:TRIGger[:SEQuence]:SINGle

Example of use

```
10 OUTPUT 717;":TRIG:SING"
20 OUTPUT 717;"*OPC?"
30 ENTER 717;A
```

SCPI.TRIGger.SEQuence.SOURce

Object type

Property (**Read-Write**)

Syntax

SCPI.TRIGger.SEQuence.SOURce = *Param*

Param = SCPI.TRIGger.SEQuence.SOURce

Description

This command sets/gets the trigger source from the following 4 types:

Trigger	Description
Internal Trigger	Uses the internal trigger to generate continuous triggers automatically.
External Trigger	Generates a trigger when the trigger signal is inputted externally through the External Trigger Input connector or the 24 Bit I/O Port (handler interface).
Manual Trigger	Generates a trigger when the key operation of **Trigger** > **Trigger** is executed from the front panel.
Bus Trigger	Generates a trigger when the SCPI.IEEE4882.TRG object is executed.

NOTE When you change the trigger source during sweep, the sweep is aborted.

Variable

Parameter	*Param*
Description	Trigger source
Data type	Character string type (String)
Range	Select from either of the following: • "INTernal": Specifies internal trigger. • "EXTernal": Specifies external trigger.

	• "MANual": Specifies manual trigger. • "BUS": Specifies bus trigger.
Preset value	"INTernal"

Examples

```
Dim TrigSour As String
SCPI.TRIGger.SEQuence.SOURce = "bus"
TrigSour = SCPI.TRIGger.SEQuence.SOURce
```

Equivalent key

Trigger > **Trigger Source** > **Internal**|**External**|**Manual**|**Bus**

Equivalent SCPI command

Syntax

```
:TRIGger[:SEQuence]:SOURce {INTernal|EXTernal|MANual|BUS}
:TRIGger[:SEQuence]:SOURce?
```

Query response

```
{BUS|EXT|INT|MAN}<newline><^END>
```

Example of use

```
10 OUTPUT 717;":TRIG:SOUR BUS"
20 OUTPUT 717;":TRIG:SOUR?"
30 ENTER 717;A$
```