PULSE Mode

For RSA6000 Series Spectrum Analyzer

Programming Guide

Mar. 2026

Guaranty and Declaration

Trademark Information
RIGOL® is the trademark of RIGOL TECHNOLOGIES CO., LTD.

Notices
- **RIGOL** products are covered by P.R.C. and foreign patents, issued and pending.
- **RIGOL** reserves the right to modify or change parts of or all the specifications and pricing policies at the company's sole decision.
- Information in this publication replaces all previously released materials.
- Information in this publication is subject to change without notice.
- **RIGOL** shall not be liable for either incidental or consequential losses in connection with the furnishing, use, or performance of this manual, as well as any information contained.
- Any part of this document is forbidden to be copied, photocopied, or rearranged without prior written approval of **RIGOL**.

Product Certification
RIGOL guarantees that this product conforms to the national and industrial standards in China as well as the ISO9001:2015 standard and the ISO14001:2015 standard. Others international standard conformance certifications are in progress

Contact Us
If you have any problem or requirement when using our products or this manual, please contact **RIGOL**.
E-mail: service@rigol.com
Website: *http://www.rigol.com*

Chapter 1 Document Overview

This manual introduces how to program and control **RIGOL** RSA6000 series spectrum analyzer (PULSE mode) by using SCPI commands through the USB or LAN interface.

Tip

For the latest version of this manual, download it from the official website of **RIGOL** (*http://www.rigol.com*).

Publication Number

PGD35100-1110

Software Version

00.01.01

Software upgrade might change or add product features. Please acquire the latest version of the manual from **RIGOL** website or contact **RIGOL** to upgrade the software.

Format Conventions in this Manual

1. **Key**

The front panel key is denoted by the menu key icon. For example, 

indicates the **System** key.

2. **Menu**

The menu item is denoted by the format of "Menu Name (Bold) + Character Shading" in the manual. For example, **Setup** indicates clicking or tapping the "Setup" sub-menu under the **System** menu to view the basic configuration items.

3. **Operation Procedures**

The next step of the operation is denoted by an arrow ">" in the manual. For example,  > **Save** indicates that first clicking or tapping the icon  , then clicking or tapping **Save**.

4. **Connector**

The connectors on the front or rear panel are usually denoted by the format of "Connector Name (Bold) + Square Brackets (Bold)". For example, **[TRIG IN]**.

Content Conventions in this Manual

The RSA6000 series spectrum analyzer includes the following models. Unless otherwise specified, this manual takes RSA6265 as an example to illustrate the functions and operation methods of RSA5065 series spectrum analyzer.

Model	Frequency Range
RSA6265	5 kHz to 26.5 GHz
RSA6140	5 kHz to 14 GHz
RSA6085	5 kHz to 8.5 GHz

Contents

Chapter 2Programming Overview

This chapter introduces how to set up remote communication between the spectrum analyzer and the PC, the remote control methods, the syntax, symbols, parameters, and abbreviation rules of the SCPI commands.

SCPI Command Overview

The remote control can be realized by using SCPI (Standard Commands for Programmable Instruments) commands. SCPI (Standard Commands for Programmable Instruments) is a standardized instrument programming language that is built upon the existing standard IEEE 488.1 and IEEE 488.2 and conforms to various standards, such as the floating point operation rule in IEEE 754 standard, ISO 646 7-bit coded character set for information interchange (equivalent to ASCII programming). The SCPI commands provide a hierarchical tree structure, and consist of multiple subsystems. Each command subsystem consists of one root keyword and one or more sub-keywords.

Syntax

The command line usually starts with a colon; the keywords are separated by colons, and following the keywords are the parameter settings available. The command ending with a quotation mark indicates querying a certain function and returns the query results. The keywords of the command and the first parameter are separated by a space.

For example,

:SYSTem:BEEPer <bool>

:SYSTem:BEEPer?

SYSTem is the root keyword of the command. BEEPer is the second-level keywords. The command line starts with a colon, and different levels of keywords are also separated by colons. <bool> indicates a settable parameter. The command ending with a quotation mark indicates querying a certain function. :SYSTem:BEEPer and the parameter <bool> are separated by a space.

In some commands with parameters, "," is often used to separate multiple parameters. For example,

:SYSTem:DATE <year>,<month>,<day>.

Symbol Description

The following symbols are not sent with the commands.

1. Braces { }

The contents in the braces can contain one or multiple parameters. These parameters can be omitted or used for several times. Parameters are usually separated by the vertical bar "|". When using the command, you must select one of the parameters.

2. Vertical Bar |

The vertical bar is used to separate multiple parameters. When using the command, you must select one of the parameters.

3. Square Brackets [ ]

The contents in the square brackets can be omitted.

4. Angle Brackets < >

The parameter enclosed in the angle brackets must be replaced by an effective value.

Parameter Type

1. Bool

The parameter can be set to ON, OFF, 1, or 0. For example,
:SYSTem:BEEPer <bool>
:SYSTem:BEEPer?
Wherein, <bool> can be set to {{1|ON}|{0|OFF}}. The query returns 1 or 0.

2. Discrete

The parameter can be any of the values listed. For example,
:SYSTem:PSTatus <status>
:SYSTem:PSTatus?
Wherein,
<status> can be set to DEFault|OPEN.
The query returns DEF or OPEN.

3. Integer

Unless otherwise specified, the parameter can be any integer (NR1 format) within the effective value range.
Note:
Do not set the parameter to a decimal, otherwise, errors will occur.

For example,
:SYSTem:GPIB <adr>
:SYSTem:GPIB?
Wherein, <adr> can be set to an integer ranging from 1 to 30. The query returns an integer ranging from 1 to 30.

4. Real

The parameter can be any real number within the effective value range, and this command accepts parameter input in decimal (NR2 format) and scientific notation (NR3 format). For example,
[:SENSe]:LPLot:SMOoth <percent>
[:SENSe]:LPLot:SMOoth?
Wherein, <percent> can be set to a real number ranging from 1% to 50%. The query returns the smoothing value in scientific notation.

5. ASCII String

The parameter can be the combinations of ASCII characters. For example,
In the command :CALCulate:LPLot:LLINe<n>:DESCription <string>, the parameter <string> can be set to
"Test Limit"

Command Abbreviation

All of the keywords of the commands are case-insensitive. They can all be in upper case or in lower case. If abbreviation is used, you must input all the capital letters in the command. For example,
[:SENSe]:LPLot:DETector?
can be abbreviated to
:SENS:LPL:DET?

Remote Control

The instrument can communicate with the PC via the USB interface and LAN interface to realize remote control of the instrument by using the SCPI (Standard Commands for Programmable Instruments) commands.

PC Software

Users usually need to use the PC software to send commands to control the instrument remotely. RIGOL Sigma is recommended. When the instrument is connected to the PC via the USB interface and USB, Ultra Sigma (only supported for Win 10 operating system and below) can be used to search the instrument resource and send the command.

Log in to the RIGOL official website. Click SUPPORT CENTER and select SOFTWARE and FIRMWARE to obtain the Ultra Sigma software package and help documentation (https://www.rigol.com/intl/support/firmware-software.html).

Web Control

When the instrument is connected to the PC via the LAN interface, you can use Web Control to send SCPI commands from the PC to the instrument. The operation procedures are as follows:

1. Obtain the instrument's IP address and input it in the browser address bar to log in to the Web Control page.
2. After you enter the Web Control interface, click the "SCPI Panel Control" button to enter the SCPI Command interface.
3. Input the specified SCPI command and then click **Send & Read** to send the command. The operation process and the returned value will be displayed in the current interface.

Remote Control via USB

1. **Connect the device**

   Use the USB cable to connect the rear-panel USB DEVICE interface of the instrument to the USB HOST interface of the PC.

2. **Search for the device resource**

   Start up Ultra Sigma and the software will automatically search for the resource currently connected to the PC via the USB interface. You can also click **USB-TMC** to search for the resource.

3. **View the device resource**

   The resources found will appear under the "RIGOL Online Resource" directory, and the model number and USB interface information of the instrument will also be displayed.

4. **Control the instrument remotely**

   Right-click the resource name and select "SCPI Panel Control" to open the remote command control panel. Then you can send commands and read data through the panel. For details about the SCPI commands and programming, refer to the Programming Guide of this instrument.

Remote Control via LAN

1. **Connect the device**

   Use the network cable to connect the instrument to your local area network (LAN).

2. **Configure network parameters**

   In **System** > **I/O** menu, configure the network parameters for the instrument.

3. **Search for the device resource**

   Start up Ultra Sigma and click **LAN** to open the panel. Click **Search** and the software searches for the instrument resources currently connected and the resources found are displayed at the right section of the window. Click **OK** to add it. Besides, you can input the IP address of the instrument manually into the text field under "Manual Input LAN Instrument IP", then click **TEST**. If the instrument passes the test, click **ADD** to add the instrument to the LAN instrument resource list in the right section; if the instrument fails the test, please check whether the IP address that you input is correct, or use the auto search method to add the instrument resource.

4. **View the device resource**

   The resources found are shown in the "RIGOL Online Resource" directory.

5. **Control the instrument remotely**

   Right-click the resource name and select "SCPI Panel Control" to open the remote command control panel. Then you can send commands and read data through the panel.

6. **Load LXI webpage**

   This instrument conforms to LXI CORE 2011 DEVICE standards, you can load LXI webpage through Ultra Sigma (right-click the instrument resource name and select "LXI-Web"). Various important information about the instrument (including the model, manufacturer, serial number, description, MAC address, and IP address) will be displayed on the webpage. You can also directly input the IP address of the instrument in the address bar of the PC browser to load the LXI webpage.

Chapter 3 Command System

This chapter introduces the commands of the RSA6000 series spectrum analyzer in PULSE mode.

Remarks:

1. The commands concerning the phase noise measurement are only available for the RSA6000 model installed with the RSA6000-PMA option. For details, refer to remarks for each command subsystem.
2. For the command set, unless otherwise specified, the query command returns "N/A" (without quotations in its return format) if no specified option is installed. If the queried function is disabled or improper type match is found, the query command will return "Error" (without quotations in its return format).
3. This manual takes RSA6265 as an example to illustrate the range of the parameters in each command.

:CALCulate Commands

The :CALCulate commands mainly cover the functions such as marker, measurement setup, and etc.

:CALCulate:PULSe:DETection:THReshold

Syntax
:CALCulate:PULSe:DETection:THReshold <real>
:CALCulate:PULSe:DETection:THReshold?

Description
Sets or queries the pulse detection threshold.

Parameter

Name	Type	Range	Default
<real>	Real	-100 to 100	-10

Remarks
- When the detection threshold reference is set to Top Level, Base Level, or Trigger Level, the pulse detection threshold is defined in dB, relative to the defined reference.
- When the detection threshold reference is set to Absolute, the pulse detection threshold is defined as an absolute threshold, expressed in dBm.

Return Format
The query returns the pulse detection threshold in scientific notation.

Example
:CALC:PULS:DET:THR -5 /*Sets the pulse detection threshold in to -5 dB.*/
:CALC:PULS:DET:THR? /*The query returns -5.0E+00.*/

:CALCulate:PULSe:DETection:THReshold:REFerence

Syntax
:CALCulate:PULSe:DETection:THReshold:REFerence <enum>
:CALCulate:PULSe:DETection:THReshold:REFerence?

Description
Sets or queries the pulse detection threshold reference.

Parameter

Name	Type	Range	Default
<enum>	Discrete	TOP|BOTTom|ABSolute|TRIGger	TOP

Remarks

TOP: indicates Top Level. By default, the pulse detection threshold reference is Top Level. It indicates setting the detection threshold level as a relative threshold offset from the top level.
BOTTom: indicates Base Level. It indicates setting the detection threshold level as a relative threshold offset from the base level.
ABSolute: indicates Absolute. It indicates setting the detection threshold level as an absolute threshold level.
TRIGger: indicates Trigger Level. It indicates setting the detection threshold level as a relative threshold offset from the IF trigger level.

Return Format
The query returns TOP, BOTTom, ABSolute, or TRIGger.

Example
:CALC:PULS:DET:THR:REF BOTTom /*Sets the pulse detection threshold reference to Base Level.*/
:CALC:PULS:DET:THR:REF? /*The query returns BOTTom.*/

:CALCulate:PULSe:DETection:THReshold:LINE

Syntax
:CALCulate:PULSe:DETection:THReshold:LINE <bool>
:CALCulate:PULSe:DETection:THReshold:LINE?

Description
Sets or queries whether to display the detection threshold line.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 0 or 1.

Example
:CALC:PULS:DET:THR:LINE ON or :CALC:PULS:DET:THR:LINE 1
/*Enables the display of the detection threshold line.*/
:CALC:PULS:DET:THR:LINE? /*The query returns 1.*/

:CALCulate:PULSe:DETection:HYSTeresis

Syntax
:CALCulate:PULSe:DETection:HYSTeresis <real>
:CALCulate:PULSe:DETection:HYSTeresis?

Description
Sets or queries the hysteresis level.

Parameter

Name	Type	Range	Default
<real>	Real	0 dB to 50 dB	1 dB

Return Format
The query returns the hysteresis level in scientific notation. The unit is dB.

Example
:CALC:PULS:DET:HYST 15 /*Sets the hysteresis level to 15 dB.*/
:CALC:PULS:DET:HYST? /*The query returns 1.5E+01.*/

:CALCulate:PULSe:DETection:MINimum

Syntax
:CALCulate:PULSe:DETection:MINimum <bool>
:CALCulate:PULSe:DETection:MINimum?

Description
Sets or queries whether to enable min. pulse width.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:DET:MIN ON or :CALC:PULS:DET:MIN 1 /*Enables Min. pulse width.*/
:CALC:PULS:DET:MIN? /*The query returns 1.*/

:CALCulate:PULSe:DETection:MINimum:WIDTh

Syntax
:CALCulate:PULSe:DETection:MINimum:WIDTh <time>
:CALCulate:PULSe:DETection:MINimum:WIDTh?

Description
Sets or queries min. pulse width.

Parameter

Name	Type	Range	Default
<time>	Discrete	0 to 512 μs	100 ns

Return Format
The query returns the min. pulse width in scientific notation. The default unit is s.

Example
:CALC:PULS:DET:MIN:WIDT 2.1E-04 /*Sets the min. pulse width to 210 μs.*/
:CALC:PULS:DET:MIN:WIDT? /*The query returns 2.1E-04.*/

:CALCulate:PULSe:DETection:MAXimum

Syntax
:CALCulate:PULSe:DETection:MAXimum <bool>
:CALCulate:PULSe:DETection:MAXimum?

Description
Sets or queries whether to enable max. pulse width.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:DET:MAX ON or :CALC:PULS:DET:MAX 1 /*Enables max. pulse width.*/
:CALC:PULS:DET:MAX? /*The query returns 1.*/

:CALCulate:PULSe:DETection:MAXimum:WIDTh

Syntax
:CALCulate:PULSe:DETection:MAXimum:WIDTh <time>
:CALCulate:PULSe:DETection:MAXimum:WIDTh?

Description
Sets or queries max. pulse width.

Parameter

Name	Type	Range	Default
<time>	Discrete	0 to max. measurement time	512 μs

Return Format
The query returns the max. pulse width in scientific notation. The default unit is s.

Example
:CALC:PULS:DET:MAX:WIDT 5.12E-04 /*Sets the max. pulse width to 512 μs.*/
:CALC:PULS:DET:MAX:WIDT? /*The query returns 5.12E-04.*/

:CALCulate:PULSe:ANALysis:THReshold:RISE:UPPer

Syntax
:CALCulate:PULSe:ANALysis:THReshold:RISE:UPPer <real>
:CALCulate:PULSe:ANALysis:THReshold:RISE:UPPer?

Description
Sets or queries the upper threshold level of rising time.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	90%

Return Format
The query returns the upper threshold level of rising time in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:THR:RISE:UPP 50 /*Sets the upper threshold level of rising time to 50%.*/
:CALC:PULS:ANAL:THR:RISE:UPP? /*The query returns 5.0E+01.*/

:CALCulate:PULSe:ANALysis:THReshold:RISE:LOWer

Syntax
:CALCulate:PULSe:ANALysis:THReshold:RISE:LOWer <real>
:CALCulate:PULSe:ANALysis:THReshold:RISE:LOWer?

Description
Sets or queries the lower threshold level of rising time.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	10%

Return Format
The query returns the lower threshold level of rising time in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:THR:RISE:LOW 30 /*Sets the lower threshold level of rising time to 30%.*/
:CALC:PULS:ANAL:THR:RISE:LOW? /*The query returns 3.0E+01.*/

:CALCulate:PULSe:ANALysis:THReshold:FALL:UPPer

Syntax
:CALCulate:PULSe:ANALysis:THReshold:FALL:UPPer <real>

:CALCulate:PULSe:ANALysis:THReshold:FALL:UPPer?

Description
Sets or queries the upper threshold level of falling time.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	90%

Return Format
The query returns the upper threshold level of falling time in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:THR:FALL:UPP 50 /*Sets the upper threshold level of falling time to 50%.*/
:CALC:PULS:ANAL:THR:FALL:UPP? /*The query returns 5.0E+01.*/

:CALCulate:PULSe:ANALysis:THReshold:FALL:LOWer

Syntax
:CALCulate:PULSe:ANALysis:THReshold:FALL:LOWer <real>
:CALCulate:PULSe:ANALysis:THReshold:FALL:LOWer?

Description
Sets or queries the lower threshold level of falling time.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	10%

Return Format
The query returns the lower threshold level of falling time in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:THR:FALL:LOW 30 /*Sets the lower threshold level of falling time to 30%.*/
:CALC:PULS:ANAL:THR:FALL:LOW? /*The query returns 3.0E+01.*/

:CALCulate:PULSe:ANALysis:THReshold:PWIDth

Syntax
:CALCulate:PULSe:ANALysis:THReshold:PWIDth <real>
:CALCulate:PULSe:ANALysis:THReshold:PWIDth?

Description

Sets or queries the pulse width threshold.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	50%

Return Format
The query returns the pulse width threshold in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:THR:PWID 60 /*Sets the pulse width threshold to 60%.*/
:CALC:PULS:ANAL:THR:PWID? /*The query returns 6.0E+01.*/

:CALCulate:PULSe:ANALysis:LEVel:METHod

Syntax
:CALCulate:PULSe:ANALysis:LEVel:METHod <enum>
:CALCulate:PULSe:ANALysis:LEVel:METHod?

Description
Sets or queries the level method.

Parameter

Name	Type	Range	Default
<enum>	Discrete	MODE|MEDian|MEAN	MODE

Remarks
MODE: Mode. It takes the pulse histogram mode levels as the top level and base level.
MEDian: Median. It takes the pulse histogram median levels as the top level and base level.
MEAN: Mean. It takes the pulse histogram mean levels as the top level and base level.

Return Format
The query returns MODE, MED, or MEAN.

Example
:CALC:PULS:ANAL:LEV:METH MEDian /*Sets the level method to MEDian.*/
:CALC:PULS:ANAL:LEV:METH? /*The query returns MED.*/

:CALCulate:PULSe:ANALysis:SPECtrum:FFT:WINDow

Syntax
:CALCulate:PULSe:ANALysis:SPECtrum:FFT:WINDow <enum>
:CALCulate:PULSe:ANALysis:SPECtrum:FFT:WINDow?

Description
Sets or queries the FFT window type.

Parameter

Name	Type	Range	Default
<enum>	Discrete	UNIForm|HANNing|FLATtop|GAUSsian	HANNing

Remarks

UNIForm: indicates the Rectangular window.
HANNing: indicates the Hanning window.
FLATtop: indicates the Flattop window.
GAUSsian: indicates the Gaussian window.

Return Format

The query returns UNIF, HANN, FLAT, or GAUS.

Example

```
:CALC:PULS:ANAL:SPEC:FFT:WIND UNIForm
/*Sets the FFT window type to Rectangular.*/
:CALC:PULS:ANAL:SPEC:FFT:WIND?              /*The query returns UNIF.*/
```

:CALCulate:PULSe:ANALysis:SPECtrum:FFT:LENGth

Syntax

```
:CALCulate:PULSe:ANALysis:SPECtrum:FFT:LENGth <integer>
:CALCulate:PULSe:ANALysis:SPECtrum:FFT:LENGth?
```

Description

Sets or queries Max. FFT length.

Parameter

Name	Type	Range	Default
<integer>	Integer	64 to 16384	16384

Return Format

The query returns the Max. FFT length in integer.

Example

```
:CALC:PULS:ANAL:SPEC:FFT:LENG 500 /*Sets the max. FFT length to 500.*/
:CALC:PULS:ANAL:SPEC:FFT:LENG?    /*The query returns 500.*/
```

:CALCulate:PULSe:ANALysis:WIDTh:RIPPle

Syntax

```
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle <real>
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle?
```

Description

Sets or queries the ripple analysis center value.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	50%

Remarks
This command is only valid when the ripple analysis reference is set to "Center".

Return Format
The query returns the ripple analysis center value in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:WIDT:RIPP 50 /*Sets the ripple analysis center value to 50%.*/
:CALC:PULS:ANAL:WIDT:RIPP? /*The query returns 5.0E+01.*/

:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:REFerence

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:REFerence <enum>
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:REFerence?

Description
Sets or queries the ripple analysis reference.

Parameter

Name	Type	Range	Default
<enum>	Discrete	CENTer|EDGE	CENTer

Return Format
The query returns CENT or EDGE.

Example
:CALC:PULS:ANAL:WIDT:RIPP:REF CENTer /*Sets the ripple analysis reference to CENTer.*/
:CALC:PULS:ANAL:WIDT:RIPP:REF? /*The query returns CENT.*/

:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:EDGE:STARt

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:EDGE:STARt <time>
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:EDGE:STARt?

Description
Sets or queries the ripple analysis edge start value.

Parameter

Name	Type	Range	Default
<time>	Real	0 s to 1 s	100 ns

Remarks
This command is only valid when the ripple analysis reference is set to "Edge".

Return Format
The query returns the ripple analysis edge start value in scientific notation. The default unit is s.

Example
:CALC:PULS:ANAL:WIDT:RIPP:EDGE:STAR 0.5 /*Sets the ripple analysis edge start value to 500 ms.*/
:CALC:PULS:ANAL:WIDT:RIPP:EDGE:STAR? /*The query returns 5.0E-01.*/

:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:EDGE:STOP

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:EDGE:STOP <time>
:CALCulate:PULSe:ANALysis:WIDTh:RIPPle:EDGE:STOP?

Description
Sets or queries the ripple analysis edge stop value.

Parameter

Name	Type	Range	Default
<time>	Real	0 s to 1 s	100 ns

Remarks
This command is only valid when the ripple analysis reference is set to "Edge".

Return Format
The query returns the ripple analysis edge stop value in scientific notation. The default unit is s.

Example
:CALC:PULS:ANAL:WIDT:RIPP:EDGE STOP 0.5 /*Sets the ripple analysis edge stop value to 500 ms.*/
:CALC:PULS:ANAL:WIDT:RIPP:EDGE:STOP? /*The query returns 5.0E-01.*/

:CALCulate:PULSe:ANALysis:WIDTh:DROop

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:DROop <real>
:CALCulate:PULSe:ANALysis:WIDTh:DROop?

Description
Sets or queries the droop analysis center value.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	100%

Remarks
This command is only valid when the droop analysis reference is set to "Center".

Return Format
The query returns the droop analysis center value in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:WIDT:DRO 85 /*Sets the droop analysis center value to 85%.*/
:CALC:PULS:ANAL:WIDT:DRO? /*The query returns 8.5E+01.*/

:CALCulate:PULSe:ANALysis:WIDTh:DROop:REFerence

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:DROop:REFerence <enum>
:CALCulate:PULSe:ANALysis:WIDTh:DROop:REFerence?

Description
Sets or queries the droop analysis reference.

Parameter

Name	Type	Range	Default
<enum>	Discrete	CENTer|EDGE	CENTer

Return Format
The query returns CENT or EDGE.

Example
:CALC:PULS:ANAL:WIDT:DRO:REF CENTer /*Sets the droop analysis reference to CENTer.*/
:CALC:PULS:ANAL:WIDT:DRO:REF? /*The query returns CENT.*/

:CALCulate:PULSe:ANALysis:WIDTh:DROop:EDGE:STARt

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:DROop:EDGE:STARt <time>
:CALCulate:PULSe:ANALysis:WIDTh:DROop:EDGE:STARt?

Description
Sets or queries the droop analysis edge start value.

Parameter

Name	Type	Range	Default
<time>	Real	0 s to 1 s	100 ns

Remarks
This command is only valid when the droop analysis reference is set to "Edge".

Return Format
The query returns the droop analysis edge start value in scientific notation. The default unit is s.

Example
:CALC:PULS:ANAL:WIDT:DRO:EDGE:STAR 0.5 /*Sets the droop analysis edge start value to 500 ms.*/
:CALC:PULS:ANAL:WIDT:DRO:EDGE:STAR? /*The query returns 5.0E-01.*/

:CALCulate:PULSe:ANALysis:WIDTh:DROop:EDGE:STOP

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:DROop:EDGE:STOP <time>
:CALCulate:PULSe:ANALysis:WIDTh:DROop:EDGE:STOP?

Description
Sets or queries the droop analysis edge stop value.

Parameter

Name	Type	Range	Default
<time>	Real	0 s to 1 s	100 ns

Remarks
This command is only valid when the droop analysis reference is set to "Edge".

Return Format
The query returns the droop analysis edge stop value in scientific notation. The default unit is s.

Example
:CALC:PULS:ANAL:WIDT:DRO:EDGE:STOP 0.5 /*Sets the droop analysis edge stop value to 500 ms.*/
:CALC:PULS:ANAL:WIDT:DRO:EDGE:STOP? /*The query returns 5.0E-01.*/

:CALCulate:PULSe:ANALysis:WIDTh:FREQuency

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:FREQuency <real>
:CALCulate:PULSe:ANALysis:WIDTh:FREQuency?

Description
Sets or queries the frequency/phase analysis center value.

Parameter

Name	Type	Range	Default
<real>	Real	0% to 100%	75%

Remarks
This command is only valid when the frequency/phase analysis reference is set to "Center".

Return Format
The query returns frequency/phase analysis center value in scientific notation. The unit is %.

Example
:CALC:PULS:ANAL:WIDT:FREQ 85 /*Sets the frequency/phase analysis center value to 85%.*/
:CALC:PULS:ANAL:WIDT:FREQ? /*The query returns 8.5E+01.*/

:CALCulate:PULSe:ANALysis:WIDTh:FREQency:REFerence

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:FREQency:REFerence <enum>
:CALCulate:PULSe:ANALysis:WIDTh:FREQency:REFerence?

Description
Sets or queries the frequency/phase analysis reference.

Parameter

Name	Type	Range	Default
<enum>	Discrete	CENTer|EDGE	CENTer

Return Format
The query returns CENT or EDGE.

Example
:CALC:PULS:ANAL:WIDT:FREQ:REF CENTer /*Sets the frequency/phase analysis reference to CENTer.*/
:CALC:PULS:ANAL:WIDT:FREQ:REF? /*The query returns CENT.*/

:CALCulate:PULSe:ANALysis:WIDTh:FREQency:EDGE:STARt

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:FREQency:EDGE:STARt <time>
:CALCulate:PULSe:ANALysis:WIDTh:FREQency:EDGE:STARt?

Description
Sets or queries the frequency/phase analysis edge start value.

Parameter

Name	Type	Range	Default

<time>	Real	0 s to 1 s	100 ns

Remarks
This command is only valid when the frequency/phase analysis reference is set to "Edge".

Return Format
The query returns the frequency/phase analysis edge start value in scientific notation. The default unit is s.

Example
:CALC:PULS:ANAL:WIDT:FREQ:EDGE:STAR 0.5 /*Sets the frequency/phase analysis edge start value to 500 ms.*/
:CALC:PULS:ANAL:WIDT:FREQ:EDGE:STAR? /*The query returns 5.0E-01.*/

:CALCulate:PULSe:ANALysis:WIDTh:FREQency:EDGE:STOP

Syntax
:CALCulate:PULSe:ANALysis:WIDTh:FREQency:EDGE:STOP <time>
:CALCulate:PULSe:ANALysis:WIDTh:FREQency:EDGE:STOP?

Description
Sets or queries the frequency/phase analysis edge stop value.

Parameter

Name	Type	Range	Default
<time>	Real	0 s to 1 s	100 ns

Remarks
This command is only valid when the frequency/phase analysis reference is set to "Edge".

Return Format
The query returns the frequency/phase analysis edge stop value in scientific notation. The default unit is s.

Example
:CALC:PULS:ANAL:WIDT:FREQ:EDGE:STOP 0.5 /*Sets the frequency/phase analysis edge stop value to 500 ms.*/
:CALC:PULS:ANAL:WIDT:FREQ:EDGE:STOP? /*The query returns 5.0E-01.*/

:CALCulate:PULSe:ANALysis:MODulation:CW

Syntax
:CALCulate:PULSe:ANALysis:MODulation:CW <bool>
:CALCulate:PULSe:ANALysis:MODulation:CW?

Description
Sets or queries whether the CW state is enabled.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:ANAL:MOD:CW ON or :CALC:PULS:ANAL:MOD:CW 1
/*Enables the CW state.*/
:CALC:PULS:ANAL:MOD:CW? /*The query returns 1.*/

:CALCulate:PULSe:ANALysis:MODulation:LFM

Syntax
:CALCulate:PULSe:ANALysis:MODulation:LFM <bool>
:CALCulate:PULSe:ANALysis:MODulation:LFM?

Description
Sets or queries whether the LFM state is enabled.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:ANAL:MOD:LFM ON or :CALC:PULS:ANAL:MOD:LFM 1
/*Enables the LFM state.*/
:CALC:PULS:ANAL:MOD:LFM? /*The query returns 1.*/

:CALCulate:PULSe:ANALysis:MODulation:BPSK

Syntax
:CALCulate:PULSe:ANALysis:MODulation:BPSK <bool>
:CALCulate:PULSe:ANALysis:MODulation:BPSK?

Description
Sets or queries whether the BPSK state is enabled.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:ANAL:MOD:BPSK ON or :CALC:PULS:ANAL:MOD:BPSK 1
/*Enables the BPSK state.*/
:CALC:PULS:ANAL:MOD:BPSK? /*The query returns 1.*/

:CALCulate:PULSe:ANALysis:MODulation:QPSK

Syntax
:CALCulate:PULSe:ANALysis:MODulation:QPSK <bool>
:CALCulate:PULSe:ANALysis:MODulation:QPSK?

Description
Sets or queries whether the QPSK state is enabled.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:ANAL:MOD:QPSK ON or :CALC:PULS:ANAL:MOD:QPSK 1
/*Enables the QPSK state.*/
:CALC:PULS:ANAL:MOD:QPSK? /*The query returns 1.*/

:CALCulate:PULSe:COUNt:MAX

Syntax
:CALCulate:PULSe:COUNt:MAX <integer>
:CALCulate:PULSe:COUNt:MAX?

Description
Sets or queries max. pulse count.

Parameter

Name	Type	Range	Default
<integer>	Integer	1 to 100	100

Return Format
The query returns the max. pulse count in integer.

Example

:CALC:PULS:COUN:MAX 80 /*Sets the max. pulse count to 80.*/
:CALC:PULS:COUN:MAX? /*The query returns 80.*/

:CALCulate:PULSe:COUNt:MAX:STATe

Syntax
:CALCulate:PULSe:COUNt:MAX:STATe <bool>
:CALCulate:PULSe:COUNt:MAX:STATe?

Description
Sets or queries whether to enable max. pulse count setting.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:COUN:MAX:STAT 1 /*Enables the max. pulse count setting.*/
:CALC:PULS:COUN:MAX:STAT? /*The query returns 1.*/

:CALCulate:PULSe:REFerence:TIME

Syntax
:CALCulate:PULSe:REFerence:TIME <enum>
:CALCulate:PULSe:REFerence:TIME?

Description
Sets or queries the time reference.

Parameter

Name	Type	Range	Default
<enum>	Discrete	FPULse|TRIGger	FPULse

Return Format
The query returns FPUL or TRIG.

Example
:CALC:PULS:REF:TIME FPULse /*Sets the time reference to First Pulse.*/
:CALC:PULS:REF:TIME? /*The query returns FPUL.*/

:CALCulate:PULSe:REFerence:FREQuency

Syntax
:CALCulate:PULSe:REFerence:FREQuency <enum>
:CALCulate:PULSe:REFerence:FREQuency?

Description
Sets or queries the frequency reference.

Parameter

Name	Type	Range	Default
<enum>	Discrete	CENTer|ABSolute	CENTer

Return Format
The query returns CENT or ABS.

Example
:CALC:PULS:REF:FREQ ABSolute /*Sets the frequency reference to ABSolute.*/
:CALC:PULS:REF:FREQ? /*The query returns ABS.*/

:CALCulate:PULSe:SELect:ALL

Syntax
:CALCulate:PULSe:SELect:ALL <bool>
:CALCulate:PULSe:SELect:ALL?

Description
Sets or queries whether to enable the "Select All" state.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:SEL:ALL 1 /*Enables the "Select All" state.*/
:CALC:PULS:SEL:ALL? /*The query returns 1.*/

:CALCulate:PULSe:SELect:STARt

Syntax
:CALCulate:PULSe:SELect:STARt <integer>
:CALCulate:PULSe:SELect:STARt?

Description
Sets or queries the start pulse.

Parameter

Name	Type	Range	Default
<integer>	Integer	1 to 100	1

Remarks
This command is only valid when "Select All" is set to "OFF".

Return Format
The query returns the start pulse in integer.

Example
:CALC:PULS:SEL:STAR 50 /*Sets the start pulse to 50.*/
:CALC:PULS:SEL:STAR? /*The query returns 50.*/

:CALCulate:PULSe:SELect:LENGth

Syntax
:CALCulate:PULSe:SELect:LENGth <integer>
:CALCulate:PULSe:SELect:LENGth?

Description
Sets or queries the number of selected pulses.

Parameter

Name	Type	Range	Default
<integer>	Integer	1 to 100	1

Remarks
This command is only valid when "Select All" is set to "OFF".

Return Format
The query returns the number of selected pulses in integer.

Example
:CALC:PULS:SEL:LENG 50 /*Sets the number of selected pulses to 50.*/
:CALC:PULS:SEL:LENG? /*The query returns 50.*/

:CALCulate:PULSe:EXTRa:INCLude

Syntax
:CALCulate:PULSe:EXTRa:INCLude <bool>
:CALCulate:PULSe:EXTRa:INCLude?

Description
Sets or queries whether to enable "Include Extra Data".

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 1 or 0.

Example
:CALC:PULS:EXTR:INCL 1 /*Enables "Include Extra Data".*/
:CALC:PULS:EXTR:INCL? /*The query returns 1.*/

:CALCulate:PULSe:MARKer:SELect

Syntax
:CALCulate:PULSe:MARKer:SELect <num>
:CALCulate:PULSe:MARKer:SELect?

Description
Sets or queries the selected marker.

Parameter

Name	Type	Range	Default
<num>	Discrete	1|2|3|4|5|6|7|8	1

Return Format
The query returns 1, 2, 3, 4, 5, 6, 7, or 8.

Example
:CALCulate:PULS:MARK:SEL 1 /*Selects Marker 1.*/
:CALCulate:PULS:MARK:SEL? /*The query returns 1.*/

:CALCulate:PULSe:MARKer:COUPle[:STATe]

Syntax
:CALCulate:PULSe:MARKer:COUPle[:STATe] <bool>
:CALCulate:PULSe:MARKer:COUPle[:STATe]?

Description
Sets or queries the marker coupling state.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Remarks
When this function enabled, moving any marker will enable other markers (except the OFF marker) to move with it.

Return Format
The query returns 0 or 1.

Example
:CALC:PULS:MARKer:COUPle:STAT OFF or :CALC:PULS:MARKer:COUPle:STAT 0
/*Disables the coupling state of the marker.*/
:CALC:PULS:MARKer:COUPle:STAT? /*The query returns 0.*/

:CALCulate:PULSe:MARKer:AOFF

Syntax
:CALCulate:PULSe:MARKer:AOFF

Description
Turns off all the enabled markers.

:CALCulate:PULSe:MARKer<n>:MODE

Syntax
:CALCulate:PULSe:MARKer<n>:MODE <mode>
:CALCulate:PULSe:MARKer<n>:MODE?

Description
Sets or queries the marker mode of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1
<mode>	Discrete	POSition|DELTa|OFF	OFF

Remarks
POSition: indicates the normal marker.
DELTa: indicates difference between two data points.
OFF: turns off the selected marker.

Return Format
The query returns POS, DELT, or OFF.

Example
:CALC:PULS:MARKer1:MODE POSition /*Sets the marker mode of Marker 1 to POSition.*/
:CALC:PULS:MARKer1:MODE? /*The query returns POS.*/

:CALCulate:PULSe:MARKer<n>:TRACe

Syntax
:CALCulate:PULSe:MARKer<n>:TRACe <integer>
:CALCulate:PULSe:MARKer<n>:TRACe?

Description
Sets or queries the marker window of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1
<integer>	Integer	1|2|3|4	1

Remarks
<Integer> indicates the marker window. It can be set to any integer ranging from 1 to 4.

Return Format
The query returns the marker window in integer, that is, 1, 2, 3 or 4.

Example
:CALC:PULS:MARKer1:TRACe 2 /*Sets the marker window of Marker 1 to Window 2.*/
:CALC:PULS:MARKer1:TRACe? /*The query returns 2.*/

:CALCulate:PULSe:MARKer<n>:X

Syntax
:CALCulate:PULSe:MARKer<n>:X <real>
:CALCulate:PULSe:MARKer<n>:X?

Description
Sets or queries the marker X value of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	——
<real>	Real	——	--

Return Format
The query returns the marker X value of the specified marker in real number or in scientific notation. The unit is related to the selected window where the marker is located.

Example
:CALC:PULS:MARKer1:X 20000 /*Sets the marker X value of Marker 1 to 20 kHz.*/
:CALC:PULS:MARKer1:X? /*The query returns 20000.*/

:CALCulate:PULSe:MARKer<n>:REFerence

Syntax
:CALCulate:PULSe:MARKer<n>:REFerence <ref_marker>

:CALCulate:PULSe:MARKer<n>:REFerence?

Description
Sets or queries the reference marker of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1
<ref_marker>	Integer	1 to 8	Next marker

Return Format
The query returns the reference marker for the specified marker in integer.

Example
:CALC:PULS:MARKer1:REFerence 2 /*Sets the reference marker of Marker 1 to Marker 2.*/
:CALC:PULS:MARKer1:REFerence? /*The query returns 2.*/

:CALCulate:PULSe:MARKer<n>:FUNCtion

Syntax
:CALCulate:PULSe:MARKer<n>:FUNCtion <func>
:CALCulate:PULSe:MARKer<n>:FUNCtion?

Description
Sets or queries the interval function of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1
<func>	Real	OFF|PCOUnt	OFF

Remarks
OFF: disables the interval function.
PCOUnt: indicates pulse count.
This command is only valid when the specified marker is located in the RawTime window.

Return Format
The query returns OFF or PCOU.

Example
:CALC:PULS:MARKer1:FUNCtion PCOUnt /*Sets the interval function of Marker 1 to Pulse Count.*/
:CALC:PULS:MARKer1:FUNCtion? /*The query returns PCOU.*/

:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:SPAN

Syntax
:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:SPAN <real>
:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:SPAN?

Description
Sets or queries the interval span of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1
<real>	Real	0 to 40 ms	20 μs

Remarks
To set or query the Interval Left value, run the **:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:LEFT** command; to set or query the Interval Right value, run the **:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:RIGHt** command.

Return Format
The query returns the interval span of the specified marker in real number. The default unit is s.

Example
:CALC:PULS:MARK1:FUNC:BAND:SPAN 20 /*Sets the interval span of Marker 1 to 20 s.*/
:CALC:PULS:MARK1:FUNC:BAND:SPAN? /*The query returns 20.*/

:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:LEFT

Syntax
:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:LEFT <real>
:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:LEFT?

Description
Sets the interval left edge of the interval span of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1
<real>	Real	0 to 40 ms	40 μs

Remarks
When the interval left value is modified, the interval span will be updated automatically.

Return Format
The query returns the interval left edge for the interval span of the specified marker in scientific notation. The default unit is s.

Example

:CALC:PULS:MARK1:FUNC:BAND:LEFT 4E-05 /*Sets the Interval Left value of Marker 1 to 40 μs.*/
:CALC:PULS:MARK1:FUNC:BAND:LEFT? /*The query returns 4E-05.*/

:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:RIGHt

Syntax
:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:RIGHt <real>
:CALCulate:PULSe:MARKer<n>:FUNCtion:BAND:RIGHt?

Description
Sets the interval right edge of the interval span of the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1
<real>	Real	0 to 40 ms	60 μs

Remarks
When the interval right value is modified, the interval span will be updated automatically.

Return Format
The query returns the interval right edge for the interval span of the specified marker in scientific notation. The default unit is s.

Example
:CALC:PULS:MARK1:FUNC:BAND:RIGH 6E-05 /*Sets the Interval Right value of Marker 1 to 60 μs.*/
:CALC:PULS:MARK1:FUNC:BAND:RIGH? /*The query returns 6E-05.*/

:CALCulate:PULSe:MARKer<n>:FUNCtion:RESult?

Syntax
:CALCulate:PULSe:MARKer<n>:FUNCtion:RESult?

Description
Queries the calculation results of the marker function for the specified marker.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1

Remarks
The query returns the calculation results of the marker function for the specified marker in integer.

Example
:CALC:PULS:MARK1:FUNCtion:RESult? /*The query returns 10.*/

:CALCulate:PULSe:MARKer<n>:MAXimum

Syntax
:CALCulate:PULSe:MARKer<n>:MAXimum

Description
Moves the specified marker to the max. peak position.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1

Example
:CALC:PULS:MARK1:MAX /*Moves Marker 1 to the max. peak position.*/

:CALCulate:PULSe:MARKer<n>:MAXimum:NEXT

Syntax
:CALCulate:PULSe:MARKer<n>:MAXimum:NEXT

Description
Moves the specified marker to the next peak.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1

Example
:CALC:PULS:MARK1:MAX:NEXT /*Moves Marker 1 to the next peak.*/

:CALCulate:PULSe:MARKer<n>:MINimum

Syntax
:CALCulate:PULSe:MARKer<n>:MINimum

Description
Moves the specified marker to the min. peak position.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4|5|6|7|8	1

Example
:CALC:PULS:MARK1:MIN /*Moves Marker 1 to the min. peak position.*/

:CALibration Commands

:CALibration[:ALL]

Syntax
:CALibration[:ALL]

Description
Executes self-calibration immediately.

Example
:CALibration:ALL /*Executes self-calibration immediately.*/

:CALibration:AUTO

Syntax
:CALibration:AUTO <bool>
:CALibration:AUTO?

Description
Enables or disables auto calibration.
Queries the setting status of auto calibration.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 0 or 1.

Example
:CALibration:AUTO ON or :CALibration:AUTO 1 /*Enables the auto calibration.*/
:CALibration:AUTO? /*The query returns 1.*/

:CONFigure Commands

:CONFigure:CATalog?

Syntax
:CONFigure:CATalog?

Description
Queries the current measurement function.

Return Format
The query returns PULSE.

:DISPlay Commands

:DISPlay:PULSe:VIEW[:SELect]

Syntax

:DISPlay:PULSe:VIEW[:SELect] <view>
:DISPlay:PULSe:VIEW[:SELect]?

Description

Sets or queries the current combination view.

Parameter

Name	Type	Range	Default
<view>	Discrete	BASIc|FREQuency|TABLe|SCATter	BASIc

Remarks

BASIc: basic view. This view includes the RawTime window and Pulse Table window.
FREQuency: Frequency view. This view includes the Amplitude, FM, and Phase windows.
TABLe: Table view. This view includes the Current Statistics and Cumulative Statistics windows.
SCATter: Scatter view. This view includes Scatter 1 window and Scatter 2 window.

Return Format

The query returns BASIc, FREQuency, TABLe, or SCATter.

Example

:DISP:PULS:VIEW BASIc /*Sets the combination view to BASIc.*/
:DISP:PULS:VIEW? /*The query returns BASIc.*/

:DISPlay:PULSe:WINDow<n>:ENABle

Syntax

:DISPlay:PULSe:WINDow<n>:ENABle <bool>
:DISPlay:PULSe:WINDow<n>:ENABle?

Description

Sets or queries whether to enable the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<bool>	Bool	ON|OFF|1|0	——

Remarks

Different traces can be set in different windows. At most four windows are available to choose on the screen at one time, and they are indicated in numeral number 1, 2, 3, and 4.

Return Format

The query returns 1 or 0.

Example
:DISP:PULS:WIND1:ENAB 1 /*Enables Window 1.*/
:DISP:PULS:WIND1:ENAB? /*The query returns 1.*/

:DISPlay:PULSe:WINDow<n>:TYPE

Syntax
:DISPlay:PULSe:WINDow<n>:TYPE <integer>
:DISPlay:PULSe:WINDow<n>:TYPE?

Description
Sets or queries the type of the trace in the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<integer>	Integer	0 to 91	——

Remarks
Different traces can be set in different windows. At most four windows are available to choose on the screen at one time, and they are indicated in numeral number 1, 2, 3, and 4. No. 0 through No.91 indicate different window types. The following table lists the window names and their corresponding window IDs.

Window ID Table

Window ID	Window Name	Window ID	Window Name
0	Raw Time	46	TrendLine Droop Stop
1	Spectrum	47	TrendLine Overshoot (dB)
2	Amplitude	48	TrendLine Overshoot (%)
3	Phase	49	TrendLine Ripple (dB)
4	FM	50	TrendLine Ripple (%)
5	Reserved	51	Histogram Top Level
6	Reserved	52	Histogram Base Level
7	Reserved	53	Histogram Top/Base
8	Spectrogram	54	Histogram On Level
9	Scatter 1	55	Histogram Peak Level
10	Scatter 2	56	Histogram Mean Level
11	Reserved	57	Histogram Peak to Average
12	Reserved	58	Histogram Rise Time
13	Reserved	59	Histogram Fall Time
14	TrendLine Top Level	60	Histogram Rise Edge
15	TrendLine Base Level	61	Histogram Fall Edge
16	TrendLine Top/Base	62	Histogram Width
17	TrendLine On Level	63	Histogram Off Time
18	TrendLine Peak Level	64	Histogram PRI
19	TrendLine Mean Level	65	Histogram PRF

Window ID	Window Name	Window ID	Window Name
20	TrendLine Peak to Average	66	Histogram Duty Cycle
21	TrendLine Rise Time	67	Histogram Phase Mean
22	TrendLine Fall Time	68	Histogram Phase Pulse-Pulse Diff
23	TrendLine Rise Edge	69	Histogram Phase Pk-Pk Dev
24	TrendLine Fall Edge	70	Histogram Phase Error RMS
25	TrendLine Width	71	Histogram Phase Error Peak
26	TrendLine Off Time	72	Histogram Phase Error Peak Loc
27	TrendLine PRI	73	Histogram Freq Mean
28	TrendLine PRF	74	Histogram Freq Pulse-Pulse Diff
29	TrendLine Duty Cycle	75	Histogram Freq Pk-Pk Dev
30	TrendLine Phase Mean	76	Histogram Freq Error RMS
31	TrendLine Phase Pulse-Pulse Diff	77	Histogram Freq Error Peak
32	TrendLine Phase Pk-Pk Dev	78	Histogram Freq Error Peak Loc
33	TrendLine Phase Error RMS	79	Histogram Droop (dB)
34	TrendLine Phase Error Peak	80	Histogram Droop (%)
35	TrendLine Phase Error Peak Loc	81	Histogram Droop Rate
36	TrendLine Freq Mean	82	Histogram Droop Start
37	TrendLine Freq Pulse-Pulse Diff	83	Histogram Droop Stop
38	TrendLine Freq Pk-Pk Dev	84	Histogram Overshoot (dB)
39	TrendLine Freq Error RMS	85	Histogram Overshoot (%)
40	TrendLine Freq Error Peak	86	Histogram Ripple (dB)
41	TrendLine Freq Error Peak Loc	87	Histogram Ripple (%)
42	TrendLine Droop (dB)	88	Demod Bits
43	TrendLine Droop (%)	89	Pulse Table
44	TrendLine Droop Rate	90	Current Statistics Table
45	TrendLine Droop Start	91	Cumulative Statistics Table

Return Format

The query returns 1 or 0.

Example

```
:DISP:PULS:WIND1:TYPE 1       /*Sets Window 1 to Spectrum window.*/
:DISP:PULS:WIND1:TYPE?    /*The query returns 1.*/
```

:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:COUPle

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:COUPle <bool>
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:COUPle?

Description
Sets or queries whether to enable the X-axis auto scaling for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<bool>	Bool	OFF|ON|0|1	ON|1

Remarks
ON|1: The X-axis auto scaling function is enabled. The X Ref Value and X Scale/Div are auto coupled. The instrument will adjust the X Ref Value and X Scale/Div value automatically to optimal state.
OFF|0: The X-axis auto scaling function is disabled. The X Ref Value and X Scale/Div are not auto coupled. You can set the X Ref Value and X Scale/Div value manually.

Return Format
The query returns 0 or 1.

Example
:DISPlay:PULS:WINDow1:TRAC:X:SCALe:COUPle ON
or :DISPlay:PULS:WINDow1:TRAC:X:SCALe:COUPle 1
/*Enables X Auto Scaling for Window 1.*/
:DISPlay:PULS:WINDow1:TRAC:X:SCALe:COUPle? /*The query returns 1.*/

:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:RLEVel

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:RLEVel <real>
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:RLEVel?

Description
Sets or queries the X Ref Value for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<real>	Real	-1e+12 to 1e+12	0

Remarks
This command is only valid when X Auto Scaling is disabled.

The unit of X Ref Value is determined by selected window type. For details, refer to "X Ref Value" section in the User Guide.

Return Format
The query returns the X Ref Value of the specified trace in real number.

Example
:DISP:PULS:WIND1:TRAC:X:SCALe:RLEVel 25 /*Sets the X Ref Value in Window 1 to 25.*/
:DISP:PULS:WIND1:TRAC:X:SCALe:RLEVel? /*The query returns 25.*/

:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:PDIVision

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:PDIVision <real>
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:PDIVision?

Description
Sets or queries the X Scale/Div value for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<real>	Real	1e-11 to 1e+11	10

Remarks
The unit of the X Scale/Div value is different for the trace in different windows. For details, refer to "X Scale/Div" section in the User Guide.

Return Format
The query returns the X Scale/Div value of the trace in the specified window in real number.

Example
:DISP:PULS:WIND1:TRAC:X:PDIV 10 /*Sets the X Scale/Div value of Window 1 to 10 dB.*/
:DISP:PULS:WIND1:TRAC:X:PDIV? /*The query returns 10.*/

:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:RPOSition

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:RPOSition <enum>
:DISPlay:PULSe:WINDow<n>:TRACe:X[:SCALe]:RPOSition?

Description
Sets or queries the X Ref Position for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<enum>	Discrete	LEFT|CENTer|RIGHt	LEFT

Return Format
The query returns LEFT, CENT, or RIGH.

Example
:DISP:PULS:WIND1:TRAC:X:SCALe:RPOSition LEFT /*Sets the X Ref Position of Window 1 to LEFT.*/
:DISP:PULS:WIND1:TRAC:X:SCALe:RPOSition? /*The query returns LEFT.*/

:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:COUPle

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:COUPle <bool>
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:COUPle?

Description
Sets or queries whether to enable the Y-axis auto scaling for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<bool>	Bool	OFF|ON|0|1	ON|1

Remarks
ON|1: The Y-axis auto scaling function is enabled. The Y Ref Value and Y Scale/Div are auto coupled. The instrument will adjust the Y Ref Value and Y Scale/Div value automatically to optimal state.
OFF|0: The Y-axis auto scaling function is disabled. The Y Ref Value and Y Scale/Div are not auto coupled. You can set the Y Ref Value and Y Scale/Div value manually.

Return Format
The query returns 0 or 1.

Example
:DISPlay:PULS:WINDow1:TRAC:Y:SCALe:COUPle ON
or :DISPlay:PULS:WINDow1:TRAC:Y:SCALe:COUPle 1
/*Enables Y Auto Scaling for Window 1.*/
:DISPlay:PULS:WINDow1:TRAC:Y:SCALe:COUPle? /*The query returns 1.*/

:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:RLEVel

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:RLEVel <real>

:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:RLEVel?

Description
Sets or queries the Y Ref Value for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<real>	Real	-1e+12 to 1e+12	0

Remarks
This command is only valid when Y Auto Scaling is disabled.
The unit of Y Ref Value is determined by selected window type. For details, refer to "Y Ref Value" section in the User Guide.

Return Format
The query returns Y Ref Value of the specified trace in real number.

Example
:DISP:PULS:WIND1:TRAC:Y:SCALe:RLEVel 25 /*Sets the Y Ref Value in Window 1 to 25.*/
:DISP:PULS:WIND1:TRAC:Y:SCALe:RLEVel? /*The query returns 25.*/

:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:PDIVision

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:PDIVision <real>
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:PDIVision?

Description
Sets or queries the Y Scale/Div value for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<real>	Real	1e-11 to 1e+11	10

Return Format
The query returns the Y Scale/Div value of the trace in the specified window in real number.

Example
:DISP:PULS:WIND1:TRAC:Y:PDIV 10 /*Sets the Y Scale/Div value of the trace in Window 1 to 10 dB.*/
:DISP:PULS:WIND1:TRAC:Y:PDIV? /*The query returns 10.*/

:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:RPOSition

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:RPOSition <enum>
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:RPOSition?

Description
Sets or queries the Y Ref Position of the trace for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<enum>	Discrete	TOP|CENTer|BOTTom	TOP

Remarks
This command is invalid for the Table window, Demod Bits window, and Spectrogram window.

Return Format
The query returns TOP, CENT, or BOTT.

Example
:DISP:PULS:WIND1:TRAC:Y:SCAL:RPOS CENTer /*Sets the Y Ref Position of the trace in Window 1 to CENTer.*/
:DISP:PULS:WIND1:TRAC:Y:SCALe:RPOS? /*The query returns CENT.*/

:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:UNIT

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:UNIT <enum>
:DISPlay:PULSe:WINDow<n>:TRACe:Y[:SCALe]:UNIT?

Description
Sets or queries the Y Axis Unit of the trace for the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<enum>	Discrete	DBM|V|W	DBM

Remarks
The Y-axis unit is only available to set in the RawTime and Amplitude windows. Its setting will affect the measurement trace, reference trace, and error trace in the Amplitude window.

Return Format
The query returns DBM, V, or W.

Example

:DISP:PULS:WIND1:TRAC:Y:SCAL:UNIT V /*Sets the Y Axis Unit of the trace in Window 1 to V.*/
:DISP:PULS:WIND1:TRAC:Y:SCAL:UNIT? /*The query returns V.*/

:DISPlay:PULSe:WINDow<n>:TRACe<tnum>[:STATe]

Syntax
:DISPlay:PULSe:WINDow<n>:TRACe<tnum>[:STATe] <bool>
:DISPlay:PULSe:WINDow<n>:TRACe<tnum>[:STATe]?

Description
Sets or queries whether to enable the specified trace type in the specified window.

Parameter

Name	Type	Range	Default
<n>	Discrete	1|2|3|4	——
<tnum>	Discrete	1|2|3	——
<bool>	Bool	OFF|ON|0|1	ON|1

Remarks
<tnum> can be set to 1, 2, or 3. 1 corresponds to measurement trace; 2 corresponds to reference trace; and 3 corresponds to error trace.
ON|1: enabled; OFF|0: disabled.
This command is only valid in the Amplitude window, Phase window, and FM window.

Return Format
The query returns 0 or 1.

Example
:DISPlay:PULS:WIND1:TRAC1 ON or :DISPlay:PULS:WIND1:TRAC1 1
/*Enables the measurement trace in Window 1.*/
: DISPlay:PULS:WIND1:TRAC1? /*The query returns 1.*/

:DISPlay:PULSe:TABLe:ENABle

Syntax
:DISPlay:PULSe:TABLe:ENABle <str>
:DISPlay:PULSe:TABLe:ENABle?

Description
Sets or queries whether to display the metric item in the pulse table.

Parameter

Name	Type	Range	Default
<str>	ASCII String	Refer to Remarks	--

Remarks
<str> is a string a characters consisting of 0 and 1. 0 indicates that it is not displayed; 1 indicates that it is displayed. You can set whether to display the specified metric item in the pulse table.

The following table lists the default display state of the metric items in the pulse table, current statistics table, and cumulative statistics table. 1 indicates, by default, it is displayed.

Table **2-1** Default Display State of the Metric Items in the Pulse Table, Current Statistics Table, and Cumulative Statistics Table

No.	Metric Item	Pulse Table	Current Statistics Table	Cumulative Statistics Table
0	Modulation Type	1	1	1
1	Top Level	1	1	1
2	Base Level	1	1	1
3	Top/Base	1	1	1
4	On Level			
5	Peak Level			
6	Mean Level			
7	Peak to Average			
8	Rise Time	1	1	1
9	Fall Time	1	1	1
10	POS	1	1	1
11	Falling Edge	1	1	1
12	Width	1	1	1
13	Off Time	1	1	1
14	PRI	1	1	1
15	Pulse Repetition Frequency			
16	Duty	1	1	1
17	Phase Mean	1	1	1
18	Phase Pulse-Pulse Diff			
19	Phase Pk-Pk Dev			
20	Phase Error RMS	1	1	1
21	Phase Error Peak			
22	Phase Error Peak Loc			

23	Freq Mean	1	1	1
24	Freq Pulse-Pulse Diff			
25	Freq Pk-Pk Dev			
26	Freq Error RMS	1	1	1
27	Freq Error Peak			
28	Freq Error Peak Loc			
29	Droop (dB)	1	1	1
30	Droop (%)			
31	Droop Rate			
32	Droop Start			
33	Droop Stop			
34	Overshoot (dB)	1	1	1
35	Overshoot (%)			
36	Ripple (dB)	1	1	1
37	Ripple (%)			
38	Rise Low Time			
39	Rise Center Time			
40	Rise High Time			
41	Fall Low Time			
42	Fall Center Time			
43	Fall High Time			
44	Rise Low Value			
45	Rise Center Value			
46	Rise High Value			
47	Fall Low Value			
48	Fall Center Value			
49	Fall High Value			

Return Format

The query returns the display state of the metric items in the pulse table in strings.

Example

:DISP:PULS:TABLe:ENABle

"1,0,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,

1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0"
/*Sets to display modulation type, not display top level, display base level…in the pulse table.*/
:DISP:PULS:TABLe:ENABle?   /*The query returns
1,0,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1
,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0*/

:DISPlay:PULSe:TABLe:STATistics:CURRent:ENABle

Syntax
:DISPlay:PULSe:TABLe:STATistics:CURRent:ENABle <str>
:DISPlay:PULSe:TABLe:STATistics:CURRent:ENABle?

Description
Sets or queries whether to display the metric item in the current statistics table.

Parameter

Name	Type	Range	Default
<str>	ASCII String	Refer to Remarks	--

Remarks
<str> is a string a characters consisting of 0 and 1. 0 indicates that it is not displayed; 1 indicates that it is displayed. You can set whether to display the specified metric item in the current statistics table.

For the settings of metric items with the specified metric No. that are displayed in the current statistics table, refer to **Table 2-1 Default Display State of the Metric Items in the Pulse Table, Current Statistics Table, and Cumulative Statistics Table**.

Return Format
The query returns the display state of the metric items in the current statistics table in strings.

Example
:DISP:PULS:TABL:CUM:ENABl
"1,0,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0"
/*Sets to display modulation type, not display top level, display base level…in the current statistics table.*/
:DISP:PULS:TABL:CUM:ENABl?   /*The query returns
1,0,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1
,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0*/

:DISPlay:PULSe:TABLe:STATistics:CUMulative:ENABle

Syntax

:DISPlay:PULSe:TABLe:STATistics:CUMulative:ENABle <str>
:DISPlay:PULSe:TABLe:STATistics:CUMulative:ENABle?

Description

Sets or queries whether to display the metric item in the cumulative statistics table.

Parameter

Name	Type	Range	Default
<str>	ASCII String	Refer to Remarks	--

Remarks

<str> is a string a characters consisting of 0 and 1. 0 indicates that it is not displayed; 1 indicates that it is displayed. You can set whether to display the specified metric item in the cumultaive statistics table.
For the settings of metric items with the specified metric No. that are displayed in the cumulative statistics table, refer to **Table 2-1 Default Display State of the Metric Items in the Pulse Table, Current Statistics Table, and Cumulative Statistics Table**.

Return Format

The query returns the display state of the metric items in the cumulative statistics table in strings.

Example

:DISP:PULS:TABL:STAT:CUM:ENABl
"1,0,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0"
/*Sets to display modulation type, not display top level, display base level…in the cumulative statistics table.*/
:DISP:PULS:TABL:CUM:ENABl?   /*The query returns
1,0,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1
,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0*/

:DISPlay:PULSe:SCATter<snum>:Y

Syntax

:DISPlay:PULSe:SCATter<snum>:Y <enum>
:DISPlay:PULSe:SCATter<snum>:Y?

Description

Sets or queries the scatterplot Y Scale of Scatter 1 or Scatter 2 window.

Parameter

Name	Type	Range	Default
<snum>	Discrete	1|2	—
<enum>	Discrete	PNUMBER|TOPL|BASEL|TOPBASE|ONLEV|PEAKL| AVGL|PEAKAVG|RTIME|FTIME|REDGE|FEDGE|WIDTH| OFFT|PRI|PRF|DUTY|PMEAN|PDIFF|PDEV|PERR| PERRPK|PERRPKL|FMEAN|FDIFF|FDEV|FERR| FERRPK|FERRPKL|DRDB|DRPER|DRRATE|DRSTART| DRSTOP|OVSH|OVSHPER|RIPPLE|RIPPLEPER	

Remarks
PNUMBER: Pulse No.; TOPL: Top Level; BASEL: Base Level; TOPBASE: Top/Base; ONLEV: On Level; PEAKL: Peak Level; AVGL: Mean Level; PEAKAVG: Peak to Average; RTIME: Rise Time; FTIME: Fall Time; REDGE: Rising Edge; FEDGE: Falling Edge; WIDTH: Width; OFFT: Off Time; PRI: pulse repetition interval; PRF: pulse repetition frequency; DUTY: Ducy Cycle; PMEAN: Phase Mean; PDIFF: Phase Pulse-Pulse Diff; PDEV: Phase Pk-Pk Dev; PERR: Phase Error RMS; PERRPK: Phase Error Peak; PERRPKL: Phase Error Peak Loc; FMEAN: Freq Mean; FDIFF: Freq Pulse-Pulse Diff; FDEV: Freq Pk-Pk Dev; FERR: Freq Error RMS; FERRPK: Freq Error Peak; FERRPKL: Freq Error Peak Loc; DRDB: Droop (dB); DRPER: Droop (%); DRRATE: Droop Rate; DRSTART: Droop Start; DRSTOP: Droop Stop; OVSH: Overshoot (dB); OVSHPER: Overshoot (%); RIPPLE: Ripple (dB); RIPPLEPER: Ripple (%).

Return Format
The query returns PNUMBER, TOPL, BASEL, TOPBASE, ONLEV, PEAKL, AVGL, PEAKAVG, RTIME, FTIME, REDGE, FEDGE, WIDTH, OFFT, PRI, PRF, DUTY, PMEAN, PDIFF, PDEV, PERR, PERRPK, PERRPKL, FMEAN, FDIFF, FDEV, FERR, FERRPK, FERRPKL, DRDB, DRPER, DRRATE, DRSTART, DRSTOP, OVSH, OVSHPER, RIPPLE, or RIPPLEPER.

Example
:DISPlay:PULSe:SCATter1:Y FMEAN
/*Sets the scatterplot Y scale of Scatter 2 window to Freq Mean.*/
:DISPlay:PULSe:SCATter1:Y? /*The query returns FMEAN.*/

:DISPlay:PULSe:SCATter<snum>:X

Syntax
:DISPlay:PULSe:SCATter<snum>:X <enum>
:DISPlay:PULSe:SCATter<snum>:X?

Description
Sets or queries the scatterplot X Scale of Scatter 1 or Scatter 2 window.

Parameter

Name	Type	Range	Default
<snum>	Discrete	1|2	—
<enum>	Discrete	PNUMBER|TOPL|BASEL|TOPBASE|ONLEV|PEAKL| AVGL|PEAKAVG|RTIME|FTIME|REDGE|FEDGE|WIDTH| OFFT|PRI|PRF|DUTY|PMEAN|PDIFF|PDEV|PERR| PERRPK|PERRPKL|FMEAN|FDIFF|FDEV|FERR| FERRPK|FERRPKL|DRDB|DRPER|DRRATE|DRSTART| DRSTOP|OVSH|OVSHPER|RIPPLE|RIPPLEPER	

Remarks

PNUMBER: Pulse No.; TOPL: Top Level; BASEL: Base Level; TOPBASE: Top/Base; ONLEV: On Level; PEAKL: Peak Level; AVGL: Mean Level; PEAKAVG: Peak to Average; RTIME: Rise Time; FTIME: Fall Time; REDGE: Rising Edge; FEDGE: Falling Edge; WIDTH: Width; OFFT: Off Time; PRI: pulse repetition interval; PRF: pulse repetition frequency; DUTY: Ducy Cycle; PMEAN: Phase Mean; PDIFF: Phase Pulse-Pulse Diff; PDEV: Phase Pk-Pk Dev; PERR: Phase Error RMS; PERRPK: Phase Error Peak; PERRPKL: Phase Error Peak Loc; FMEAN: Freq Mean; FDIFF: Freq Pulse-Pulse Diff; FDEV: Freq Pk-Pk Dev; FERR: Freq Error RMS; FERRPK: Freq Error Peak; FERRPKL: Freq Error Peak Loc; DRDB: Droop (dB); DRPER: Droop (%); DRRATE: Droop Rate; DRSTART: Droop Start; DRSTOP: Droop Stop; OVSH: Overshoot (dB); OVSHPER: Overshoot (%); RIPPLE: Ripple (dB); RIPPLEPER: Ripple (%).

Return Format

The query returns PNUMBER, TOPL, BASEL, TOPBASE, ONLEV, PEAKL, AVGL, PEAKAVG, RTIME, FTIME, REDGE, FEDGE, WIDTH, OFFT, PRI, PRF, DUTY, PMEAN, PDIFF, PDEV, PERR, PERRPK, PERRPKL, FMEAN, FDIFF, FDEV, FERR, FERRPK, FERRPKL, DRDB, DRPER, DRRATE, DRSTART, DRSTOP, OVSH, OVSHPER, RIPPLE, or RIPPLEPER.

Example

```
:DISPlay:PULSe:SCATter1:X FMEAN
/*Sets the scatterplot Y scale of Scatter 1 window to Freq Mean.*/
:DISPlay:PULSe:SCATter1:X?    /*The query returns FMEAN.*/
```

:FETCh Commands

:FETCh:PULSe<dnum>?

Syntax

:FETCh:PULSe<dnum>?

Description

Queries the measurement results of the pulse measurement mode.

Parameter

Name	Type	Range	Default

<dnum>	Integer	Refer to **"Remarks"**	——

Remarks

The following table lists the relationship between the parameter <dnum> and the returned value to be queried.

Parameter <dnum>	Return Value	Parameter <dnum>	Return Value
0	Raw IQ Data Pair	9	Reference Time-Domain Phase
1 or default	Valid Number of Pulses	10	Error Time-Domain Phase
	Average Power	11	FM Measurement Time-Domain
	Peak Power	12	FM Reference Time-Domain
	Top Level	13	FM Error Time-Domain
	Base Level	14	Reserved
	Top/Base Level	15	Reserved
	Detect Threshold	16	Reserved
	Time Offset	17	Reserved
2	Raw Time	18	Reserved
3	Spectrum	19	Reserved
4	Reserved	20	Demod Bits Length
5	Measurement Time-Domain Amplitude	21	Demod Bits
6	Reference Time-Domain Amplitude	22	Reserved
7	Error Time-Domain Amplitude	23	Reserved
8	Measurement Time-Domain Phase	-	-

IEEE 488.2 Common Commands

IEEE 488.2 common commands are used to operate or query the status registers.

*CLS

Syntax

*CLS

Description

Clears all the event registers and status byte registers.

*ESE

Syntax
*ESE <value>
*ESE?

Description
Sets or queries the enable register of the standard event status register.

Parameter

Name	Type	Range	Default
<value>	Integer	Refer to "**Remarks**"	0

Remarks
The bit 2, bit 3, bit 4, and bit 7 are reserved; you can set their values but they will not affect the system. Bit 1 and bit 6 are not used and are always treated as 0; therefore, the range of <value> are the decimal numbers corresponding to the binary numbers ranging from 00000000 (0 in decimal) to 11111111 (255 in decimal) and of which bit 1 and bit 6 are 0.

Return Format
The query returns an integer. The integer equals to the binary-weighted sum of all the bits set in the register. For example, the query returns 144 if Bit 4 (16 in decimal) and Bit 7 (128 in decimal) are enabled.

Example
The following command sets the enable register for the standard event status register to 16.
*ESE 16

The following query returns 16.
*ESE?

*ESR?

Syntax
*ESR?

Description
Queries and clears the event register for the standard event status register.

Remarks
Bit 1 and Bit 6 in the standard event status register are not in use, and are regarded as 0. The query returns a decimal value that corresponds to the binary values ranging from 00000000 (0 in decimal) to 11111111 (255 in decimal) and of which Bit 1 and Bit 6 are 0.

Return Format
The query returns an integer. The integer equals to the binary-weighted sum of all the bits set in the register. For example, the query returns 144 if Bit 4 (16 in decimal) and Bit 7 (128 in decimal) are enabled.

Example
The following query returns 24 (Bit 3 and Bit 4 have been set).
*ESR?

*IDN?

Syntax
*IDN?

Description
Queries the ID string of instrument.

Return Format
The query returns the ID string in the following format:
RIGOL TECHNOLOGIES,<model>,<serial number>,XX.XX.XX
<model>: instrument model
<serial number>: serial number of the instrument
XX.XX.XX: software version of the instrument

Example
The following query returns RIGOL TECHNOLOGIES,RSA6265,RSA60004000000,00.00.25.
*IDN?

*OPC

Syntax
*OPC
*OPC?

Description
Sets Bit 0 (Operation Complete, OPC) in the standard event status register to 1 after the current operation is finished.
Queries whether the current operation is finished.

Return Format
The query returns 1 after the current operation is finished; otherwise, the query returns 0.

*RCL

Syntax
*RCL <integer>

Description
Recalls the selected register.

Parameter

Name	Type	Range	Default

<integer>	Integer	1 to 16	——

Example
The following command recalls Register 1.
*RCL 1

*RST

Syntax
*RST

Description
Restores the instrument to its factory default settings.

*SAV

Syntax
*SAV <integer>

Description
Saves the current instrument state to the selected register.

Parameter

Name	Type	Range	Default
<integer>	Integer	1 to 16	——

Example
The following command saves the current instrument state to Register 1.
*SAV 1

*SRE

Syntax
*SRE <value>
*SRE?

Description
Sets or queries the enable register of the status byte register.

Parameter

Name	Type	Range	Default
<value>	Integer	Refer to "**Remarks**"	0

Remarks
Bit 0 and Bit 1 are not used and are always treated as 0; therefore, the range of <value> are the decimal numbers corresponding to the binary numbers ranging from 00000000 (0 in

decimal) to 11111111 (255 in decimal) and of which Bit 0 and Bit 1 are 0.

Return Format
The query returns an integer. The integer equals to the binary-weighted sum of all the bits set in the register. For example, the query returns 144 if Bit 4 (16 in decimal) and Bit 7 (128 in decimal) are enabled.

Example
The following command sets the enable register for the status byte register to 16.
*SRE 16

The following query returns 16.
*SRE?

*STB?

Syntax
*STB?

Description
Queries the event register for the status byte register.

Remarks
Bit 0 and Bit 1 in the status byte register are not in use, and are regarded as 0. The query returns a decimal value that corresponds to the binary values ranging from 00000000 (0 in decimal) to 11111111 (255 in decimal) and of which Bit 0 and Bit 1 are 0.

Return Format
The query returns an integer. The integer equals to the binary-weighted sum of all the bits set in the register. For example, the query returns 144 if Bit 4 (16 in decimal) and Bit 7 (128 in decimal) are enabled.

Example
The following query returns 24 (Bit 3 and Bit 4 have been set).
*STB?

*TST?

Syntax
*TST?

Description
Queries whether the self-check operation is finished.

Remarks
The query returns 0 or 1. A zero is returned if the test is successful, 1 if it fails.

*WAI

Syntax
***WAI**

Description
Waits for all the pending operations to complete before executing any additional commands.

:GPIB:PARSe:END

Syntax
:GPIB:PARSe:END

Description
The GPIB module command. This instruction set sends the "Parse End" command to the GPIB module. If this command is not sent, the USB-GPIB adapter module fails to work. You will receive no return value after sending a command.

:INITiate Commands

:INITiate:CONTinuous

Syntax
:INITiate:CONTinuous <bool>
:INITiate:CONTinuous?

Description
Selects continuous (ON|1) or single (OFF|0) measurement mode.
Queries the current measurement mode.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format
The query returns 0 or 1.

Example
:INITiate:CONTinuous ON or :INITiate:CONTinuous 1 /*Performs continuous measurement.*/
:INITiate:CONTinuous? /*The query returns 1.*/

:INITiate[:IMMediate]

Syntax
:INITiate[:IMMediate]

Description
Initializes one sweep in the non-measurement state.
Triggers one measurement in the measurement state.

:INITiate:RESTart

Syntax
:INITiate:RESTart

Description
Restarts to initiate a sweep.

:INSTrument Commands

:INSTrument:COUPle:FREQuency:CENTer

Syntax

:INSTrument:COUPle:FREQuency:CENTer <enum>
:INSTrument:COUPle:FREQuency:CENTer?

Description

Sets or queries whether the global CF mode is enabled.

Parameter

Name	Type	Range	Default
<enum>	Discrete	ALL|NONE	NONE

Remarks

NONE: disables the global CF mode.
ALL: enables the global CF mode.
If you execute this command in any mode, the center frequency of the current mode is set to the global center frequency. Adjusting the center frequency in a mode, while the global center frequency is on, will modify the global center frequency.

Return Format

The query returns ALL or NONE.

Example

```
:INSTrument:COUPle:FREQuency:CENTer ALL    /*Enables the global CF mode.*/
:INSTrument:COUPle:FREQuency:CENTer?      /*The query returns ALL.*/
```

:INSTrument:SCReen:CATalog?

Syntax

:INSTrument:SCReen:CATalog?

Description

Queries the available measurement modes displayed at the bottom of the screen.

:INSTrument:SCReen:CREate

Syntax

:INSTrument:SCReen:CREate

Description

Creates a new measurement mode label at the bottom of the screen.

:INSTrument:SCReen:DELete

Syntax
:INSTrument:SCReen:DELete

Description
Deletes the currently selected measurement mode label at the bottom of the screen.

:INSTrument:SCReen:DELete:ALL

Syntax
:INSTrument:SCReen:DELete:ALL

Description
Deletes all the currently not selected measurement mode label at the bottom of the screen.

:INSTrument:SCReen:SELect

Syntax
:INSTrument:SCReen:SELect <name>
:INSTrument:SCReen:SELect?

Description
Switches to the specified screen.
Queries the currently selected screen.

Parameter

Name	Type	Range	Default
<name>	ASCII String	The label can contain English letters and numbers, as well as some symbols.	-

Remarks
You can run the ***:INSTrument:SCReen:CATalog?*** command to query the measurement mode label name.

Example
:INSTrument:SCReen:SELect PULSE 1 /*Switches to the screen PULSE 1.*/
:INSTrument:SCReen:SELect? /*The query returns "PULSE 1"/

:INSTrument[:SELect]

Syntax
:INSTrument[:SELect] <enum>
:INSTrument[:SELect]?

Description

Selects the working mode of the instrument.
Queries the working mode of the instrument.

Parameter

Name	Type	Range	Default
<enum>	Discrete	SA|GPSA_TG|RTSA|RTSA_UFS|GPSA_ACP|GPSA_CNR|GPSA_CHPower|GPSA_TOI|GPSA_HARModist|GPSA_OBW|GPSA_TPOWer|EMI|VSA|AM|FM|PM|PNOISE|PULSE|VSA_ZIGBEE|VSA_LORA|VSA_TX|GPSA_IBEM	SA

Remarks
After running the command of switching the working mode, we recommend you set the timeout value to 8 s, or perform the next operation after a delay of 8 s.

Example
:INSTrument:SELect PULSE /*Sets the instrument to work in PULSE mode.*/
:INSTrument:SELect? /*The query returns PULSE.*/

:LAN Commands

:LAN:DHCP

Syntax
:LAN:DHCP <bool>
:LAN:DHCP?

Description
Enables or disables the DHCP configuration; queries the on/off status of DHCP configuration.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Remarks
When the three IP configuration types (DHCP, Auto IP, and Static IP) are all turned on, the priority of the parameter configuration from high to low is "DHCP", "Auto IP", and "Static IP".
The three IP configuration types cannot be all turned off at the same time.
In DHCP mode, the DHCP server on the current network assigns network parameters (such as the IP address) to the instrument.
After the :LAN:APPLy command is executed, the configuration type can take effect immediately.

Return Format
The query returns 0 or 1.

Example
:LAN:DHCP OFF /*Disables DHCP configuration.*/

:LAN:DHCP? /*The query returns 0.*/

:LAN:AUToip

Syntax
:LAN:AUToip <bool>
:LAN:AUToip?

Description
Enables or disables the Auto IP configuration; queries the on/off status of Auto IP configuration.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Remarks
When the auto IP mode is valid, disable DHCP manually. In this mode, the instrument acquires the IP address ranging from "169.254.0.1" to "169.254.255.254" and the subnet mask (255.255.0.0) automatically according to the current network configuration.

Return Format
The query returns 0 or 1.

Example
:LAN:AUToip OFF /*Disables Auto IP configuration.*/
:LAN:AUToip /*The query returns 0.*/

:LAN:MANual

Syntax
:LAN:MANual <bool>
:LAN:MANual?

Description
Enables or disables the static IP configuration; queries the on/off status of static IP configuration.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Remarks
When the static IP mode is valid, disable DHCP and Auto IP manually. You can self-define the network parameters of the instrument, such as IP address, subnet mask, gateway, and DNS address. For the setting of the IP address, refer to the :LAN:IPADdress command. For the setting of the subnet mask, refer to the :LAN:SMASk command. For the setting of the

gateway, refer to the :LAN:GATeway command. For the setting of the DNS address, refer to the :LAN:DNS command.

Return Format
The query returns 0 or 1.

Example
:LAN:MANual ON /*Enables the static IP configuration.*/
:LAN:MANual? /*The query returns 1.*/

:LAN:IPADdress

Syntax
:LAN:IPADdress <string>
:LAN:IPADdress?

Description
Sets or queries the instrument IP address.

Parameter

Name	Type	Range	Default
<string>	ASCII String	Refer to "**Remarks**"	——

Remarks
The format of the<string> is "nnn.nnn.nnn.nnn". The range for the first segment (nnn) of the address is from 0 to 223 (except 127); and the range for the other three segments is from 0 to 255.
Only when the IP configuration mode is static IP and both DHCP and Auto IP are disabled, can you use the set the IP address.

Return Format
The query returns the current IP address in strings.

Example
:LAN:IPADdress 192.168.1.10 /*Sets the IP address to 192.168.1.10*/
:LAN:IPADdress? /*The query returns 192.168.1.10*/

:LAN:SMASk

Syntax
:LAN:SMASk <string>
:LAN:SMASk?

Description
Sets or queries the subnet mask.

Parameter

Name	Type	Range	Default

<string>	ASCII String	Refer to "**Remarks**"	——

Remarks
The format of <string> is "nnn.nnn.nnn.nnn". The range for the first segment (nnn) is from 0 to 255.
Only when the IP configuration mode is static IP and both DHCP and Auto IP are disabled, can you use the set subnet mask.

Return Format
The query returns the current subnet mask in strings.

Example
:LAN:SMASk 255.255.255.0 /*Sets the subnet mask to 255.255.255.0.*/
:LAN:SMASK? /*The query returns 255.255.255.0.*/

:LAN:GATeway

Syntax
:LAN:GATeway <string>
:LAN:GATeway?

Description
Sets or queries the default gateway.

Parameter

Name	Type	Range	Default
<string>	ASCII String	Refer to "**Remarks**"	——

Remarks
The format of the<string> is "nnn.nnn.nnn.nnn". The range for the first segment (nnn) of the address is from 0 to 223 (except 127); and the range for the other three segments is from 0 to 255.
Only when the IP configuration mode is static IP and both DHCP and Auto IP are disabled, can you use the set default gateway.

Return Format
The query returns the current gateway in strings.

Example
:LAN:GATeway 192.168.1.1 /*Sets the default gateway to 192.168.1.1.*/
:LAN:GATeway? /*The query returns 192.168.1.1.*/

:LAN:DNS

Syntax
:LAN:DNS <string>
:LAN:DNS?

Description
Sets or queries the DNS address.

Parameter

Name	Type	Range	Default
<string>	ASCII String	Refer to "**Remarks**"	——

Remarks
The format of the<string> is "nnn.nnn.nnn.nnn". The range for the first segment (nnn) of the address is from 0 to 223 (except 127); and the range for the other three segments is from 0 to 255.
Only when the IP configuration mode is static IP and both DHCP and Auto IP are disabled, can you use the set DNS.

Return Format
The query returns the current DNS address in strings.

Example
:LAN:DNS 192.168.1.1 /*Sets the DNS address to 192.168.1.1.*/
:LAN:DNS? /*The query returns 192.168.1.1.*/

:LAN:MAC?

Syntax
:LAN:MAC?

Description
Queries the MAC address of the instrument.

Return Format
The query returns the MAC address in strings. For example, 00:19:AF:00:11:22.

:LAN:DSERver?

Syntax
:LAN:DSERver?

Description
Queries the DHCP server address.

Return Format
The query returns the DHCP server address in strings.

:LAN:STATus?

Syntax
:LAN:STATus?

Description
Queries the current network configuration status.

Remarks
- UNLINK: Not connected!
- CONNECTED: Connected successfully.
- INIT: acquiring IP address.
- IPCONFLICT: IP conflicted.
- BUSY: please wait...
- CONFIGURED: network configuration succeeded.
- DHCPFAILED: DHCP configuration failed.
- INVALIDIP: invalid IP.
- IPLOSE: IP lost.

Return Format
The query returns UNLINK, CONNECTED, INIT, IPCONFLICT, BUSY, CONFIGURED, DHCPFAILED, INVALIDIP, or IPLOSE.

:LAN:VISA?

Syntax
:LAN:VISA? [<type>]

Description
Queries the VISA address of the instrument.

Parameter

Name	Type	Range	Default
<type>	Discrete	{USB|LXI}	——

Remarks
This command contains a parameter "type" and it is used to set or query the address type. By default, it returns the LXI address.

Return Format
The query returns the VISA address in strings.

:LAN:MDNS

Syntax
:LAN:MDNS <bool>
:LAN:MDNS?

Description
Enables or disables mDNS; or queries the mDNS status.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 0 or 1.

Example
:LAN:MDNS ON /*Enables mDNS.*/
:LAN:MDNS? /*The query returns 1.*/

:LAN:APPLy

Syntax
:LAN:APPLy

Description
Applies the network configuration.

Remarks
After configuring the LAN parameters, run this command to apply the LAN settings.

:LAN:HOST:NAME

Syntax
:LAN:HOST:NAME <name>
:LAN:HOST:NAME?

Description
Sets or queries the host name.

Parameter

Name	Type	Range	Default
<name>	ASCII String	The label can contain English letters and numbers, as well as some symbols.	——

Return Format
The query returns the host name in ASCII strings.

:LAN:DESCription

Syntax
:LAN:DESCription <name>
:LAN:DESCription?

Description

Sets or queries the description.

Parameter

Name	Type	Range	Default
<name>	ASCII String	The label can contain English letters and numbers, as well as some symbols.	——

Return Format
The query returns the description in ASCII strings.

:LOAD Commands

:LOAD:MEASure

Syntax
:LOAD:MEASure <sel>,<file_name>

Description
Loads the specified type of measurement data.

Parameter

Name	Type	Range	Default
<sel>	Discrete	RAWDATA|PULTABLE	RAWDATA
<file_name>	ASCII String	——	——

Remarks
<sel> indicates the measurement type.
RAWDATA: indicates that the measurement type is Raw Data.
PULTABLE: indicates that the measurement type is Pulse Table.
This operation fails if the file with the specified filename does not exist.

Example
:LOAD:MEASure RAWDATA,mydata /*Loads the raw data measurement file named mydata.csv.*/

Example
:MMEMory:STORe:TRACe:DATA TRACE1,mydata
/*Saves the Trace1 measurement data as a file with the file name mydata.csv to the default path (/pulse/measdata).*/

:MMEMory Commands

:MMEMory:STORe:STATe

Syntax
:MMEMory:STORe:STATe <file_name>

Description
Saves the current instrument state as a file with the specified file name suffixed with "*.sta" to the default path (/pulse/state).

Parameter

Name	Type	Range	Default
<file_name>	ASCII String	——	——

Remarks
If the specified file already exists, overwrite it.

Example
:MMEMory:STORe:STATe state /*Saves the current instrument state as a file named state.sta to the /pulse/state folder.*/

:MMEMory:STORe:SCReen

Syntax
:MMEMory:STORe:SCReen <file_name>

Description
Saves the current screenshot as a file with the specified file name suffixed with "*.jpg", "*.png", or "*.bmp" to the default path (/pulse/screen).

Parameter

Name	Type	Range	Default
<file_name>	ASCII String	——	——

Remarks
If the specified file already exists, overwrite it.
If a suffix (.jpg/.png/.bmp) is added to the filename, you can save the current screen image with a different format based on its different suffix.
If no suffix is added to the filename, then by default, the current screen image is saved in the currently selected format.

Example
:MMEMory:STORe:SCReen screen.jpg
/*Saves the current screenshot as a file with the file name screen.jpg to the /pulse/screen folder.*/

:MMEMory:STORe:IMG:POSition

Syntax
:MMEMory:STORe:IMG:POSition <position>
:MMEMory:STORe:IMG:POSition?

Description

Sets or queries the screenshot path where the specified screenshot file is saved.

Parameter

Name	Type	Range	Default
<position>	Discrete	LOCAL|OUT	LOCAL

Remarks
LOCAL: saves the screenshot to the local path.
OUT: saves the screenshot to the external storage device.

Example
:MMEMory:STORe:IMG:POSition LOCAL /*Saves the screenshot file to the local path.*/
:MMEMory:STORe:IMG:POSition? /*The query returns LOCAL.*/

:MMEMory:STORe:SCReen:DATA?

Syntax
:MMEMory:STORe:SCReen:DATA?

Description
Saves the screen data.

:MMEMory:STORe:STATe?

Syntax
:MMEMory:STORe:STATe?

Description
Obtains the status register.

:MMEMory:LOAD:STATe

Syntax
:MMEMory:LOAD:STATe <file_name>

Description
Loads the specified state file suffixed with "*.sta".

Parameter

Name	Type	Range	Default
<file_name>	ASCII String	——	——

Remarks
This operation fails if the file with the specified filename does not exist.

Example

:MMEMory:LOAD:STATe state1.sta /*Loads the state file named state1.sta to the instrument.*/

:MMEMory:USER:STORe

Syntax

:MMEMory:USER:STORe <user>
:MMEMory:USER:STORe?

Description

Sets or queries the preset file.

Parameter

Name	Type	Range	Default
<user>	Discrete	DEF|USER1|USER2|USER3|USER4|USER5|USER6	DEF

Example

:MMEMory:USER:STORe USER1 /*Sets the preset file to USER1.*/
:MMEMory:USER:STORe? /*The query returns USER1.*/

:SAVE Commands

:SAVE:MEASure

Syntax

:SAVE:MEASure <sel>,<file_name>

Description

Saves the specified type of measurement data as a file with the specified file name suffixed with "*.csv" to the default path (/pulse/measdata).

Parameter

Name	Type	Range	Default
<sel>	Discrete	RAWDATA|PULTABLE|STATABLE|CUMSTATABLE	RAWDATA
<file_name>	ASCII String	——	——

Remarks

<sel> indicates the measurement type.
RAWDATA: indicates that the measurement type is Raw Data; PULTABLE: indicates that the measurement type is Pulse Table; STATABLE: indicates that the measurement type is current statistics table; CUMSTATABLE: indicates that the measurement type is cumulative statistics table.
If the specified file already exists, overwrite it.

Example

:SAVE:MEASure RAWDATA,mydata /*Saves the raw data measurement file named mydata.csv to the default path (/pulse/measdata).*/

[:SENSe] Commands

[:SENSe]:FREQuency:CENTer

Syntax
[:SENSe]:FREQuency:CENTer <freq>
[:SENSe]:FREQuency:CENTer?

Description
Sets or queries the center frequency.

Parameter

Name	Type	Range	Default
<freq>	Real	(Smin/2)[2] to (Fmax - Smin/2)	Fmax[1]/2

Note[1]: Fmax is determined by the instrument model. RSA6000 includes 8.5 GHz model, 14 GHz model, and 26.5 GHz model.
Note[2]: Smin indicates the minimum span.

Return Format
The query returns the center frequency in real number. The unit is Hz.

Example
:SENSe:FREQuency:CENTer 5000000 /*Sets the center frequency to 5 MHz.*/
:SENSe:FREQuency:CENTer? /*The query returns 5000000.000000.*/

[:SENSe]:FREQuency:CENTer:STEP[:INCRement]

Syntax
[:SENSe]:FREQuency:CENTer:STEP[:INCRement] <freq>
[:SENSe]:FREQuency:CENTer:STEP[:INCRement]?

Description
Sets or queries the CF step.

Parameter

Name	Type	Range	Default
<freq>	Real	-Fmax to Fmax	1 MHz

Return Format
The query returns the CF step in real number. The unit is Hz.

Example
:SENSe:FREQuency:CENTer:STEP:INCRement 100000 /*Sets the CF step to 100 kHz.*/

:SENSe:FREQuency:CENTer:STEP:INCRement? /*The query returns 100000.000000.*/

[:SENSe]:FREQuency:CENTer:STEP:AUTO

Syntax

[:SENSe]:FREQuency:CENTer:STEP:AUTO <bool>
[:SENSe]:FREQuency:CENTer:STEP:AUTO?

Description

Enables or disables the auto setting mode of the CF step.
Queries the status of the auto setting mode of the CF step.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	ON|1

Return Format

The query returns 0 or 1.

Example

:SENSe:FREQuency:CENTer:STEP:AUTO ON
or :SENSe:FREQuency:CENTer:STEP:AUTO 1 /*Sets the CF step mode to Auto.*/
:SENSe:FREQuency:CENTer:STEP:AUTO? /*The query returns 1.*/

[:SENSe]:POWer[:RF]:RANGe

Syntax

[:SENSe]:POWer[:RF]:RANGe <real>
[:SENSe]:POWer[:RF]:RANGe?

Description

Sets or queries the attenuation range.

Parameter

Name	Type	Range	Default
<real>	Integer	-15 dBm to 25 dBm	0 dBm

Return Format

The query returns the attenuation range in integer. The unit is dBm.

Example

:SENSe:POWer:RF:RANG 20 /*Sets the attenuation range to 20 dB.*/
:SENSe:POWer:RF:ATTenuation? /*The query returns 20.*/

[:SENSe]:CORRection:IMPedance[:INPut][:MAGNitude]

Syntax
[:SENSe]:CORRection:IMPedance[:INPut][:MAGNitude] <enum>
[:SENSe]:CORRection:IMPedance[:INPut][:MAGNitude]?

Description
Sets or queries the input impedance. The unit is Ω.

Parameter

Name	Type	Range	Default
<enum>	Discrete	50|75	50

Remarks
If the output impedance of the system under test is 75 Ω, you should use a 75 Ω to 50 Ω adapter (option) supplied by RIGOL to connect the analyzer with the system under test, and then set the input impedance to 75 Ω.

Return Format
The query returns 50 or 75.

Example
:SENSe:CORRection:IMPedance:INPut:MAGNitude 75 /*Sets the input impedance to 75 Ω.*/
:SENSe:CORRection:IMPedance:INPut:MAGNitude? /*The query returns 75.*/

[:SENSe]:CORRection:SA[:RF]:GAIN

Syntax
[:SENSe]:CORRection:SA[:RF]:GAIN <rel_ampl>
[:SENSe]:CORRection:SA[:RF]:GAIN?

Description
Sets or queries the external gain.

Parameter

Name	Type	Range	Default
<rel_ampl>	Real	-120 dB to 120 dB	0 dB

Return Format
The query returns the external gain value in scientific notation. The unit is dB.

Example
:SENSe:CORRection:SA:RF:GAIN 20 /*Sets the external gain to 20 dB.*/
:SENSe:CORRection:SA:RF:GAIN? /*The query returns 2.0E +01.*/

[:SENSe]:PULSe:TIME:LENGth

Syntax

[:SENSe]:PULSe:TIME:LENGth <time>
[:SENSe]:PULSe:TIME:LENGth?

Description
Sets or queries the pulse measurement time.

Parameter

Name	Type	Range	Default
<time>	Real	1 μs to 40 ms	100 μs

Return Format
The query returns the pulse measurement time in scientific notation. The default unit is s.

Example
:SENSe:PULSe:TIME:LENGth 0.01 /*Sets the pulse measurement time to 10 ms.*/
:SENSe:PULSe:TIME:LENGth? /*The query returns 1.0E-02.*/

[:SENSe]:PULSe:SRATe

Syntax
[:SENSe]:PULSe:SRATe <freq>
[:SENSe]:PULSe:SRATe?

Description
Sets or queries the pulse measurement sample rate.

Parameter

Name	Type	Range	Default
<freq>	Real	12.7875 MHz to 255.75 MHz	12.7875 MHz

Return Format
The query returns the pulse measurement sample rate in scientific notation. Its unit is Hz.

Example
:SENSe:PULSe:SRATe 15000000 /*Sets the pulse measurement sample rate to 15 MHz.*/
:SENSe:PULSe:SRATe? /*The query returns 1.5E+07.*/

:SYSTem Commands

:SYSTem:BEEPer

Syntax
:SYSTem:BEEPer <bool>
:SYSTem:BEEPer?

Description

Enables or disables the beeper; queries the on/off status of the beeper.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 0 or 1.

Example
:SYSTem:BEEPer ON or :SYSTem:BEEPer 1 /*Enables the beeper.*/
:SYSTem:BEEPer? /*The query returns 1.*/

:SYSTem:BRIGhtness

Syntax
:SYSTem:BRIGhtness <brightness>
:SYSTem:BRIGhtness?

Description
Sets or queries the screen brightness.

Parameter

Name	Type	Range	Default
<brightness>	Integer	0% to 100%	80%

Return Format
The query returns the screen brightness in integer.

Example
:SYSTem:BRIGhtness 50 /*Sets the screen brightness to 50%.*/
:SYSTem:BRIGhtness? /*The query returns 50.*/

:SYSTem:DATE

Syntax
:SYSTem:DATE <year>,<month>,<day>
:SYSTem:DATE?

Description
Sets or queries the system date of the instrument.

Parameter

Name	Type	Range	Default
<year>	ASCII String	2000 to 2099	——
<month>	ASCII String	01 to 12	——
<day>	ASCII String	01 to 31	——

Return Format
The query returns the current system date in the format of "YYYY-MM-DD".

Example
:SYSTem:DATE 2017,11,16 /*Sets the system date of the instrument to Nov. 16th, 2017.*/
:SYSTem:DATE? /*The query returns 2017-11-16.*/

:SYSTem:GPIB

Syntax
:SYSTem:GPIB <adr>
:SYSTem:GPIB?

Description
Sets or queries the GPIB address.

Parameter

Name	Type	Range	Default
<adr>	Integer	1 to 30	1

Return Format
The query returns an integer ranging from 1 to 30.

Example
:SYSTem:GPIB 2 /*Sets the GPIB address to 2.*/
:SYSTem:GPIB? /*The query returns 2.*/

:SYSTem:LANGuage

Syntax
:SYSTem:LANGuage <enum>

Description
Sets or queries the system language of the instrument.

Parameter

Name	Type	Range	Default
<enum>	Discrete	SCHinese|TCHinese|KORean|JAPanese|ENGLish| GERMan|PORTuguese|POLish|FRENch|RUSSian| SPAN|THAI|INDonesian	ENGLish

Remarks
SCHinese: Simplified Chinese.
TCHinese: Traditional Chinese.
KORean: Korean.
JAPanese: Japanese.
ENGLish: English.

GERMan: German.
PORTuguese: Portuguese.
POLish: Polish.
FRENch: French.
RUSSian: Russian.
SPAN: Spanish.
THAI: Thai.
INDonesian: Indonesian.

Return Format
The query returns SCH, TCH, KOR, JAP, ENGL, GERM, PORT, POL, FREN, RUSS, SPAN, THAI, or IND.
Example
:SYSTem:LANGuage ENGLish /*Sets the system language to ENGLish.*/
:SYSTem:LANGuage? /*The query returns ENGL.*/

:SYSTem:LOCKed

Syntax
:SYSTem:LOCKed<bool>
:SYSTem:LOCKed?

Description
Locks or unlocks the keypad.
Queries whether the keypad is locked or not.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 0 or 1.

Example
:SYSTem:LOCKed 1 or :SYSTem:LOCKed ON /*Disables the touch screen operation.*/
:SYSTem:LOCKed? /*The query returns 1.*/

:SYSTem:OPTion:INSTall

Syntax
:SYSTem:OPTion:INSTall <option info>@<license info>

Description
Installs and activates the specified option.

Parameter

Name	Type	Range	Default
<option info>	ASCII String	——	--

<license info>	ASCII String	——	--

Remarks
The parameter <option info> indicates the order number of the option. <license info> indicates the serial number of the option.

Example
:SYSTem:OPTion:INSTall RSA6000-PA@8AD12B8EBC5DF492D1D4289B7CBA5B6150BF6F5D752D645C36D74530B05F39B49C461B23A50D6C94A34E06782AC4380070B0D1A86BA84E02768391FFD70C2103
/*Installs the option RSA6000-PA.*/

:SYSTem:OPTion:STATus?

Syntax
:SYSTem:OPTion:STATus? <option name>

Description
Queries whether the specified option is activated.

Parameter

Name	Type	Range	Default
<option name>	Discrete	BND|PA|P08|P14|P26|B200|RB200|EMI|T08|ADM|AWK|VSA|UBW|PNM|PMA|LORA|ZIGBEE|BLE_TX	--

Return Format
The query returns 0 (not activated) or 1 (activated).

Example
:SYSTem:OPTion:STATus? RSA6000-PA /*Queries whether the RSA6000-PA option is activated.*/

:SYSTem:OPTion:UNINstall

Syntax
:SYSTem:OPTion:UNINstall

Description
Uninstalls all the official options.

:SYSTem:PON

Syntax
:SYSTem:PON <power_on>
:SYSTem:PON?

Description

Sets or queries the setting type when the instrument is powered on.

Parameter

Name	Type	Range	Default
<power_on>	Discrete	DEFault|LATest	PRESet

Remarks
DEFault: indicates preset settings, including factory mode and 6 user-defined settings.
LATest: last settings.

Return Format
The query returns DEF or LAT.

Example
:SYSTem:PON LAT /*Sets the instrument to recall last settings at next power-on.*/
:SYSTem:PON? /*The query returns LAT.*/

:SYSTem:RESet

Syntax
:SYSTem:RESet

Description
Restarts the instrument.

:SYSTem:PRESet

Syntax
:SYSTem:PRESet

Description
Initializes the instrument.

:SYSTem:PSTatus

Syntax
:SYSTem:PSTatus <status>
:SYSTem:PSTatus?

Description
Sets or queries the front-panel power switch status.

Parameter

Name	Type	Range	Default
<status>	Discrete	DEFault|OPEN	DEFault

Remarks
DEFault: switches off. OPEN: switches on.

Return Format
The query returns DEF or OPEN.

Example
:SYSTem:PSTatus OPEN /*Sets the power switch to be on.*/
:SYSTem:PSTatus? /*The query returns OPEN.*/

:SYSTem:PWRD

Syntax
:SYSTem:PWRD

Description
Powers off the instrument.

:SYSTem:STIME

Syntax
:SYSTem:STIME <bool>
:SYSTem:STIME?

Description
Sets or queries whether to display the system time.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 0 or 1.

Example
:SYSTem:STIME 1 or :SYSTem:STIME ON /*Sets to display the system date.*/
:SYSTem:STIME? /*The query returns 1.*/

:SYSTem:TIME

Syntax
:SYSTem:TIME <hour>,<minute>,<second>
:SYSTem:TIME?

Description

Sets or queries the system time of the instrument.

Parameter

Name	Type	Range	Default
<hour>	ASCII String	00 to 23	——
<minute>	ASCII String	00 to 59	——
<second>	ASCII String	00 to 59	——

Return Format
The query returns the current system time in the format of "HH:MM:SS".

Example
:SYSTem:TIME 15,10,30 /*Sets the system time of the instrument to 15:10:30.*/
:SYSTem:TIME? /*The query returns 15:10:30.*/

:SYSTem:VERSion?

Syntax
:SYSTem:VERSion?

Description
Queries the version number of the SCPI used by the system.

:TRIGger Commands

:TRIGger[:SEQuence]:SOURce

Syntax
:TRIGger[:SEQuence]:SOURce <type>
:TRIGger[:SEQuence]:SOURce?

Description
Set or queries the trigger source.

Parameter

Name	Type	Range	Default
<type>	Discrete	IMMediate|EXTernal1|POWer	IMMediate

Remarks
IMMediate: indicates the free-run trigger.
EXTernal1: indicates the external trigger.
POWer: indicates the IF power trigger.

Return Format
The query returns IMM, EXT1, or POW.

Example

:TRIGger:SEQuence:SOURce EXTernal1 /*Sets the trigger source to External.*/
:TRIGger:SEQuence:SOURce? /*The query returns EXT1.*/

:TRIGger[:SEQuence]:ATRigger

Syntax
:TRIGger[:SEQuence]:ATRigger <time>
:TRIGger[:SEQuence]:ATRigger?

Description
Set or queries the Auto trigger time.

Parameter

Name	Type	Range	Default
<time>	Real	1 ms to 100 s	100 ms

Remarks
This command is only valid when the auto triggering function is enabled.

Return Format
The query returns the time value in scientific notation. The unit is s.

Example
:TRIGger:SEQuence:ATRigger 10 /*Sets the auto trigger time to 10 s.*/
:TRIGger:SEQuence:ATRigger? /*The query returns 1.0E+01.*/

:TRIGger[:SEQuence]:ATRigger:STATe

Syntax
:TRIGger[:SEQuence]:ATRigger:STATe <bool>
:TRIGger[:SEQuence]:ATRigger:STATe?

Description
Enables or disables the auto trigger function.
Queries the setting status of auto trigger function.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 1 or 0.

Example
:TRIGger:SEQuence:ATRigger:STATe ON or :TRIGger:SEQuence:ATRigger:STATe 1
/*Enables Auto trigger.*/
:TRIGger:SEQuence:ATRigger:STATe? /*The query returns 1.*/

:TRIGger[:SEQuence]:EXTernal1:DELay

Syntax
:TRIGger[:SEQuence]:EXTernal1:DELay <time>
:TRIGger[:SEQuence]:EXTernal1:DELay?

Description
Sets or queries the delay time for external trigger.

Parameter

Name	Type	Range	Default
<time>	Real	-500 ms to 500 ms	1 μs

Remarks
This command is only valid when the external trigger delay function is enabled.

Return Format
The query returns the delay time for the external trigger in scientific notation. The unit is s.

Example
:TRIGger:SEQuence:EXTernal1:DELay 0.1 /*Sets the delay time of the external trigger to 100 ms.*/
:TRIGger:SEQuence:EXTernal1:DELay? /*The query returns 10.0E-02.*/

:TRIGger[:SEQuence]:EXTernal1:DELay:STATe

Syntax
:TRIGger[:SEQuence]:EXTernal1:DELay:STATe <bool>
:TRIGger[:SEQuence]:EXTernal1:DELay:STATe?

Description
Enables or disables the external trigger delay function.
Queries the setting state of the external trigger delay function.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 1 or 0.

Example
:TRIGger:SEQuence:EXTernal1:DELay:STATe ON
or :TRIGger:SEQuence:EXTernal1:DELay:STATe 1
/*Enables the external trigger delay function.*/
:TRIGger:SEQuence:EXTernal1:DELay:STATe? /*The query returns 1.*/

:TRIGger[:SEQuence]:EXTernal1:SLOPe

Syntax

:TRIGger:SEQuence:EXTernal1:SLOPe <type>
:TRIGger:SEQuence:EXTernal1:SLOPe?

Description

Sets or queries the trigger edge of the external trigger.

Parameter

Name	Type	Range	Default
<type>	Discrete	POSitive|NEGative	POSitive

Remarks

POSitive: indicates the rising edge.
NEGative: indicates the falling edge.

Return Format

The query returns POS or NEG.

Example

:TRIGger:SEQuence:EXTernal1:SLOPe POSitive /*Sets the trigger edge of the external trigger to rising edge.*/
:TRIGger:SEQuence:EXTernal1:SLOPe? /*The query returns POS.*/

:TRIGger[:SEQuence]:IF:DELay

Syntax

:TRIGger[:SEQuence]:IF:DELay <time>
:TRIGger[:SEQuence]:IF:DELay?

Description

Sets or queries the IF power trigger delay time.

Parameter

Name	Type	Range	Default
<time>	Real	-500 ms to 500 ms	1 μs

Remarks

This command is only valid when the IF power trigger function is enabled.

Return Format

The query returns the delay time for IF power trigger in scientific notation. The unit is s.

Example

:TRIGger:SEQuence:IF:DELay 0.1 /*Sets the delay time of the IF power trigger to 100 ms.*/
:TRIGger:SEQuence:IF:DELay? /*The query returns 1.0E-01.*/

:TRIGger[:SEQuence]:IF:DELay:STATe

Syntax
:TRIGger[:SEQuence]:IF:DELay:STATe <bool>
:TRIGger[:SEQuence]:IF:DELay:STATe?

Description
Sets or queries whether the IF power trigger delay is enabled.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 1 or 0.

Example
:TRIGger:SEQuence:IF:DELay:STATe ON or :TRIGger:SEQuence:IF:DELay:STATe 1
/*Enables IF power trigger delay.*/
:TRIGger:SEQuence:IF:DELay:STATe? /*The query returns 1.*/

:TRIGger[:SEQuence]:IF:LEVel

Syntax
:TRIGger[:SEQuence]:IF:LEVel <time>
:TRIGger[:SEQuence]:IF:LEVel?

Description
Sets or queries the IF power trigger level.

Parameter

Name	Type	Range	Default
<time>	Real	(-140+Level Offset) to (30+Level Offset)	-25 dBm

Remarks
This command is only valid when the IF power trigger function is enabled.

Return Format
The query returns the trigger level for IF power trigger in scientific notation.

Example
:TRIGger:SEQuence:IF:LEVel 10 /*Sets the IF power trigger level to 10 dBm.*/
:TRIGger:SEQuence:IF:LEVel? /*The query returns 1.0E+01.*/

:TRIGger[:SEQuence]:HOLDoff

Syntax

:TRIGger[:SEQuence]:HOLDoff <time>
:TRIGger[:SEQuence]:HOLDoff?

Description
Sets or queries the trigger holdoff time.

Parameter

Name	Type	Range	Default
<time>	Real	0 μs to 500 ms	100 ms

Remarks
This command is only valid when the trigger holdoff function is enabled.

Return Format
The query returns the trigger holdoff time in scientific notation. The unit is s.

Example
:TRIGger:SEQuence:HOLDoff 0.1 /*Sets the holdoff time of the trigger to 100 ms.*/
:TRIGger:SEQuence:HOLDoff? /*The query returns 10.0E-02.*/

:TRIGger[:SEQuence]:HOLDoff:STATe

Syntax
:TRIGger[:SEQuence]:HOLDoff:STATe <bool>
:TRIGger[:SEQuence]:HOLDoff:STATe?

Description
Enables or disables the trigger holdoff function.
Queries the status of the trigger holdoff function.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 1 or 0.

Example
:TRIGger:SEQuence:HOLDoff:STATe ON or :TRIGger:SEQuence:HOLDoff:STATe 1
/*Enables the trigger holdoff function.*/
:TRIGger:SEQuence:HOLDoff:STATe? /*The query returns 1.*/

:TRIGger[:SEQuence]:OUTPut

Syntax
:TRIGger[:SEQuence]:OUTPut <bool>
:TRIGger[:SEQuence]:OUTPut?

Description
Sets or queries whether to enable the trigger output.

Parameter

Name	Type	Range	Default
<bool>	Bool	OFF|ON|0|1	OFF|0

Return Format
The query returns 1 or 0.

Example
:TRIGger:SEQuence:OUTPut ON or :TRIGger:SEQuence:OUTPut 1 /*Enables the trigger output.*/
:TRIGger:SEQuence:OUTPut? /*The query returns 1.*/

:TRIGger[:SEQuence]:OUTPut:POLarity

Syntax
:TRIGger[:SEQuence]:OUTPut:POLarity <type>
:TRIGger[:SEQuence]:OUTPut:POLarity?

Description
Sets or queries the trigger output polarity.

Parameter

Name	Type	Range	Default
<type>	Discrete	POSitive|NEGative	POSitive

Remarks
POSitive: positive polarity.
NEGative: negative polarity.

Return Format
The query returns POS or NEG.

Example
:TRIGger:SEQuence:OUTPut:POLarity POSitive /*Sets the trigger output polarity to POSitive.*/
:TRIGger:SEQuence:OUTPut:POLarity? /*The query returns POS.*/

Chapter 4 Programming Examples

This chapter lists some programming examples to illustrate how to use commands to realize the common functions of the spectrum analyzer in the development environments such as Visual C++ 6.0, Visual Basic 6.0, and LabVIEW 2010. Also, the chapter lists some examples to illustrate how to control the spectrum analyzer to realize the common functions in Linux operating system. These examples are programmed based on NI-VISA library.

NI-VISA (National Instrument-Virtual Instrument Software Architecture), developed by NI (National Instrument), provides an advanced programming interface to communicate with various instruments through their bus lines. NI-VISA enables you to realize the communication between the analyzer and PC through instrument buses (such as USB). VISA defines a set of software commands, with which users can control the instrument without knowing the working principle of the interface buses. For details, refer to the help information of NI-VISA.

Programming Instructions

This section introduces the problems that might occur during the programming process as well as their solutions. If these problems occur, please resolve them according to the corresponding instructions.

1. When you build a working environment via the network, it is recommended that you build a pure local area network.

2. If the local area network environment is complicated (e.g. many devices and broadcast messages exist), it is recommended that you add some fault tolerance during the programming process. For the details, refer to the instrument write/read operations with exception handling functions "InstrWriteEx()" and "InstrReadEx()" in "Visual C++ 6.0 Programming Example".

3. The Socket programming port No. of this device is 5025.

Programming Preparations

The programming preparations introduced here are only applicable to programming by using Visual C++ 6.0, Visual Basic 6.0, and LabVIEW 2010 development tools in Windows operating system. For the preparations of programming in Linux operating system, refer to "Linux Programming Example" in "Programming Preparations".

First, check whether your PC has installed NI's VISA library. If not, download it from *http://www.ni.com/visa/*. In this manual, the default installation path is C:\Program Files\IVI Foundation\VISA.
Connect spectrum analyzer to the PC via the USB interface of the analyzer. Use the USB cable to connect the USB DEVICE interface on the rear panel of the analyzer to the USB interface of the PC.

After the analyzer is connected to the PC properly, start the analyzer. In this case, "Found New Hardware Wizard" dialog box appears on the PC. Please install "USB Test and Measurement Device (IVI)" according to the wizard.

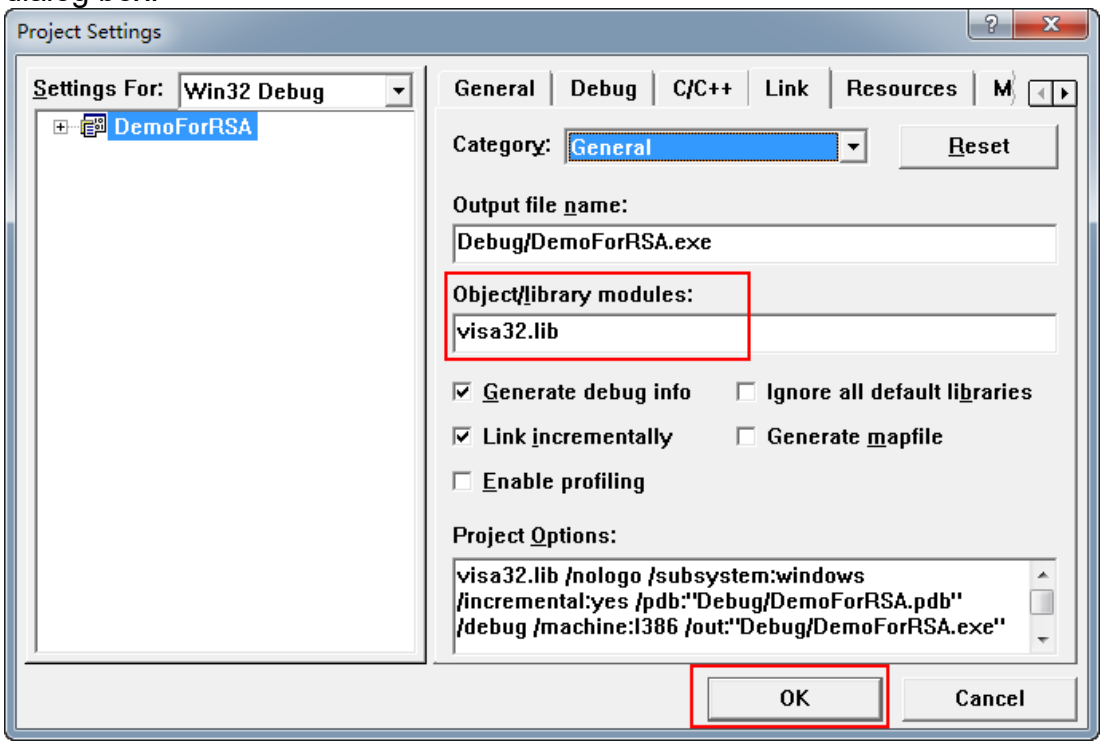
By now, the programming preparations are complete. The following parts will make a detailed introduction about the programming instances in the Visual C++ 6.0, Visual Basic 6.0, and LabVIEW 2010 development environment.

Visual C++ 6.0 Programming Example

Enter the Visual C++6.0 programming environment, and perform the following procedures.

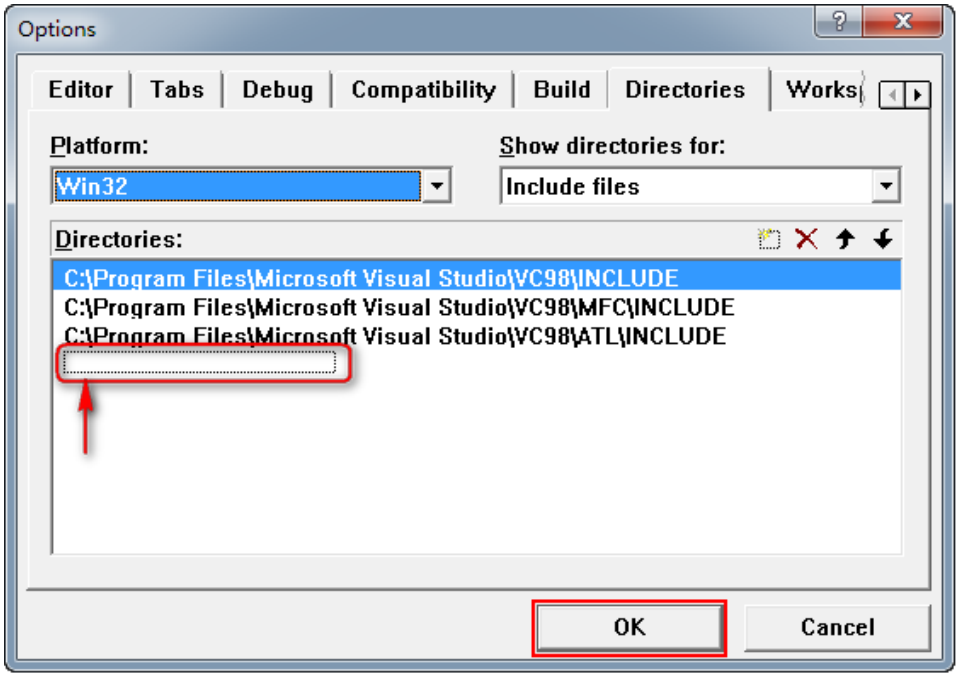
1. Create a MFC project based on a dialog box and name it "DemoForDSA" in this example.

2. Click **Project** → **Settings** to open the Project Setting dialog box. In the dialog box, click the **Link** tab, add "visa32.lib" under **Object/library modules**, then click **OK** to close the dialog box.



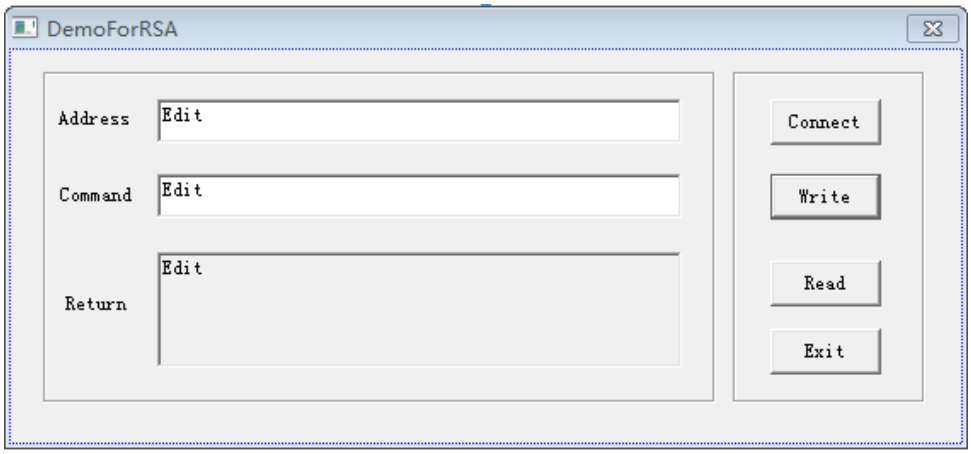
3. Click **Tools** → **Options** to open the Options dialog box. Then click the **Directories** tab. Select Include files from the drop-down list under **Show directories for**. Double click the empty space under **Directories** to enter the specified path of Include files: C:\Program Files\IVI Foundation\VISA\WinNT\include. Click **OK** to close the dialog box. Select Library files from the drop-down list under Show directories for. Double click the empty space under Directories to enter the specified path of Library files：C:\Program Files\IVI Foundation\VISA\WinNT\lib\msc. Click OK to close the dialog box.

   Note: The two paths added here are related to the installation path of NI-VISA on your PC. By default, NI-VISA is installed under C:\Program Files\IVI Foundation\VISA.



By now, VISA library has been added.

4. Add the **Text**, **Edit**, and **Button** controls. The layout interface for adding controls is as follows:



5. Add the control variables.
   Click **View → ClassWizard**, and then click on the "Member Variables" tab to add the following three variables:
   Instrument address: CString m_strInstrAddr
   Command: CString m_strCommand
   Returned value: CString m_strResult

6. Encapsulate the read and write operations of VISA.
   1) Encapsulate the write operation of VISA for easier operation.

```
bool CDemoForRSADlg::InstrWrite(CString strAddr, CString strContent) //Write operation
 {
   ViSession defaultRM,instr;
   ViStatus status;
```

```
ViUInt32 retCount;
char * SendBuf = NULL;
char * SendAddr = NULL;
bool bWriteOK = false;
CString str;

// Change the address's data style from CString to char*
SendAddr = strAddr.GetBuffer(strAddr.GetLength());
strcpy(SendAddr,strAddr);
strAddr.ReleaseBuffer();

// Change the command's data style from CString to char*
SendBuf = strContent.GetBuffer(strContent.GetLength());
strcpy(SendBuf,strContent);
strContent.ReleaseBuffer();

//Open a VISA resource
status = viOpenDefaultRM(&defaultRM);
if (status < VI_SUCCESS)
{
      AfxMessageBox("No VISA resource was opened!");
      return false;
 }

status = viOpen(defaultRM, SendAddr, VI_NULL, VI_NULL, &instr);

//Write command to the instrument
status = viWrite(instr, (unsigned char *)SendBuf, strlen(SendBuf), &retCount);

//Close the system
status = viClose(instr);
status = viClose(defaultRM);

return bWriteOK;
}
```

2) Encapsulate the read operation of VISA for easier operation.

```
bool CDemoForRSADlg::InstrRead(CString strAddr, CString *pstrResult) //Read operation
{
ViSession defaultRM,instr;
ViStatus status;
ViUInt32 retCount;
char * SendAddr = NULL;
unsigned char RecBuf[MAX_REC_SIZE];
bool bReadOK = false;
CString str;

// Change the address's data style from CString to char*
SendAddr = strAddr.GetBuffer(strAddr.GetLength());
```

```
    strcpy(SendAddr,strAddr);
    strAddr.ReleaseBuffer();

    memset(RecBuf,0,MAX_REC_SIZE);

    //Open a VISA resource
    status = viOpenDefaultRM(&defaultRM);
    if (status < VI_SUCCESS)
    {
        // Error Initializing VISA...exiting
        AfxMessageBox("No VISA resource was opened!");
        return false;
     }

    //Open the instrument
    status = viOpen(defaultRM, SendAddr, VI_NULL, VI_NULL, &instr);

    //Read from the instrument
    status = viRead(instr, RecBuf, MAX_REC_SIZE, &retCount);

    //close the system
    status = viClose(instr);
    status = viClose(defaultRM);

    (*pstrResult).Format("%s",RecBuf);

    return bReadOK;
  }
```

3) Encapsulate the read operation with exception handling function of VISA.

```
ViStatus CDemoForRSADlg::OpenVisaDevice(CString strAddr) //Open a VISA device
{
    ViStatus status;
    char * SendAddr = NULL;

    // Change the address's data style from CString to char*
    SendAddr = strAddr.GetBuffer(strAddr.GetLength());
    strcpy(SendAddr,strAddr);
    strAddr.ReleaseBuffer();

    //Open a VISA resource
    status = viOpenDefaultRM(&m_SessRM);

    if (status == 0)
    {
        //Open the device
        status = viOpen(m_SessRM, SendAddr, VI_NULL, VI_NULL,
&m_SessInstr);
```

```
            //If you fails to open the connection, close the resource
            if (status != 0)
            {
                viClose(m_SessRM);
             }
         }

        return status;
    }


    ViStatus CDemoForRSADlg::CloseVisaDevice()        //Close a VISA device

    {
        ViStatus status;

        //Close the device
        status = viClose(m_SessInstr);

        if (status == 0)
        {
            //close the resource
            status = viClose(m_SessRM);
        }

        return status;
    }


    bool CDemoForRSADlg::InstrWriteEx(CString strAddr, CString strContent) //Write
    operation with exception handling
    {
        ViStatus status;
        ViUInt32 retCount;
        char * SendBuf = NULL;
        bool bWriteOK = true;

        // Change the address's data style from CString to char*
        SendBuf = strContent.GetBuffer(strContent.GetLength());
        strcpy(SendBuf,strContent);
        strContent.ReleaseBuffer();

        do
        {
            //Write command to the instrument
            status = viWrite(m_SessInstr, (unsigned char *)SendBuf, strlen(SendBuf),
&retCount);

            //If an error occurs, perform error handing
            if (status < 0)
```

```
            {
                //If the time exceed the limit value, resend the command after a delay of
1s
                if (VI_ERROR_TMO == status)
                {
                    Sleep(1000);
                    status = viWrite(m_SessInstr, (unsigned char *)SendBuf,
                    strlen(SendBuf), &retCount);
                }
                else
                {
                    //If another error occurs, reopen the connection after the connection
                    is closed and resend the command
                    status = CloseVisaDevice();
                    Sleep(1000);
                    status = OpenVisaDevice(m_strInstrAddr);
                    if (status == 0)
                    {
                        status = viWrite(m_SessInstr, (unsigned char *)SendBuf,
                        strlen(SendBuf), &retCount);
                    }
                }
            }
        } while (status < 0);

        return bWriteOK;
    }


    bool CDemoForRSADlg::InstrReadEx(CString strAddr, CString *pstrResult) //Read
    operation with exception handling
    {
        ViStatus status;
        ViUInt32 retCount;
        char * SendAddr = NULL;
        unsigned char RecBuf[MAX_REC_SIZE];
        bool bReadOK = true;

        // Change the address's data style from CString to char*
        SendAddr = strAddr.GetBuffer(strAddr.GetLength());
        strcpy(SendAddr,strAddr);
        strAddr.ReleaseBuffer();

        memset(RecBuf,0,MAX_REC_SIZE);

        do
        {
            //Read from the instrument
            status = viRead(m_SessInstr, RecBuf, MAX_REC_SIZE, &retCount);
            if (status < 0)
```

```
                {
                    //If the time exceeds the limit value, read from the instrument after a
delay of 1s
                    if (VI_ERROR_TMO == status)
                    {
                        Sleep(1000);
                        status = viRead(m_SessInstr, RecBuf, MAX_REC_SIZE,
&retCount);
                    }
                    else
                    {
                        //If another error occurs, reopen the connection after the connection
                        is closed and reread from instrument
                        status = CloseVisaDevice();
                        Sleep(1000);
                        status = OpenVisaDevice(m_strInstrAddr);
                        if (status == 0)
                        {
                            status = viRead(m_SessInstr, RecBuf, MAX_REC_SIZE,
&retCount);
                        }
                    }
                }

            } while (status < 0);


            (*pstrResult).Format("%s",RecBuf);

            return bReadOK;
        }
```

7. Add the control message response codes.

1) Connect to the instrument

```
        void CDemoForRSADlg::OnBtConnectInstr()              // Connect to the
instrument
        {
        //TODO: Add your control notification handler code here
        ViStatus status;
        ViSession defaultRM;
        ViString expr = "?*";
        ViPFindList findList = new unsigned long;
        ViPUInt32 retcnt = new unsigned long;
        ViChar instrDesc[1000];
        CString strSrc = "";
        CString strInstr = "";
        unsigned long i = 0;
        bool bFindRSA = false;

        status = viOpenDefaultRM(&defaultRM);
```

```
        if (status < VI_SUCCESS)
        {
        // Error Initializing VISA...exiting
        MessageBox("No VISA instrument was opened ! ");
        return ;
        }

        memset(instrDesc,0,1000);

        // Find resource
        status = viFindRsrc(defaultRM,expr,findList, retcnt, instrDesc);

        for (i = 0;i < (*retcnt);i++)
        {
            // Get instrument name
            strSrc.Format("%s",instrDesc);
            InstrWrite(strSrc,"*IDN?");
            ::Sleep(200);
            InstrRead(strSrc,&strInstr);

          // If the instrument(resource) belongs to the RSA series then jump out //from the
loop
             strInstr.MakeUpper();
             if (strInstr.Find("RSA") >= 0)
             {
                 bFindRSA = true;
                 m_strInstrAddr = strSrc;
                 break;
             }

            //Find next instrument
            status = viFindNext(*findList,instrDesc);
        }

        if (bFindRSA == false)
        {
           MessageBox("Didn't find any RSA!");
        }
        UpdateData(false);
    }
```

2) Write Operation

```
    void CDemoForRSADlg::OnBtWrite()                    //Write operation
    {
    //TODO: Add your control notification handler code here
    UpdateData(true);
    if (m_strInstrAddr.IsEmpty())
    {
        MessageBox("Please connect to the instrument first!");
     }
```

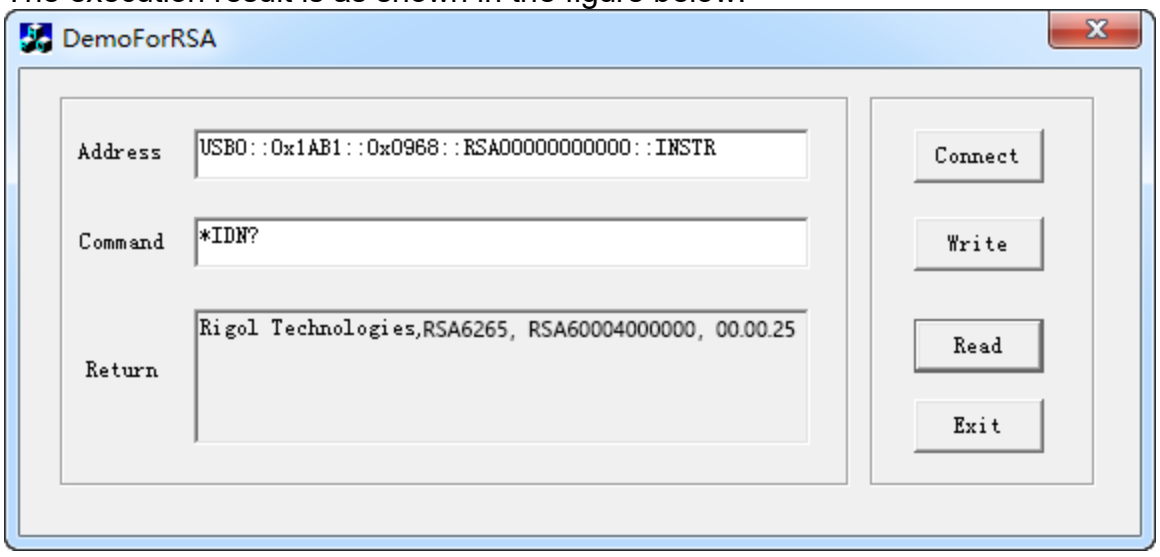
```
    InstrWrite(m_strInstrAddr,m_strCommand);
    m_strResult.Empty();
    UpdateData(false);
  }
```

3) Read Operation

```
  void CDemoForRSADlg::OnBtRead()                    //Read operation
 {
  //TODO: Add your control notification handler code here
  UpdateData(true);
  InstrRead(m_strInstrAddr,&m_strResult);
  UpdateData(false);
 }
```

8. Run the results.
   1) Click **Connect** to search for the spectrum analyzer;
   2) Input "*IDN?" in the "Command" edit box;
   3) Click **Write** to write the command into the spectrum analyzer;
   4) Click **Read** to read the return value.

The execution result is as shown in the figure below.

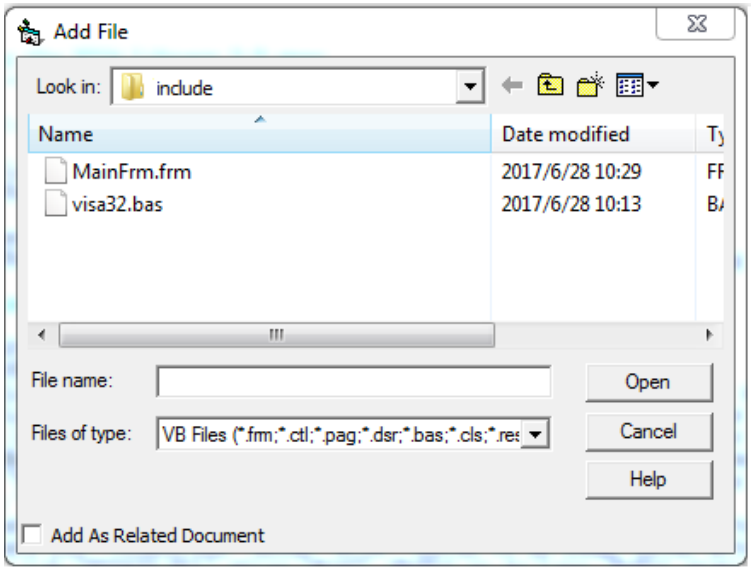


Visual Basic 6.0 Programming Example

Enter the Visual Basic 6.0 programming environment, and perform the following procedures.

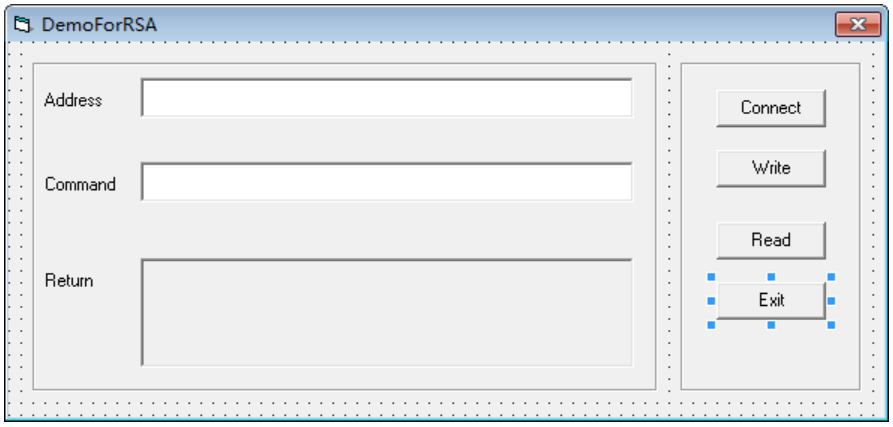
1. Build a standard application program project (Standard EXE), and name it "DemoForRSA".

2. Open **Project → Add File…**. Search for the visa32.bas file from the include folder in the installation path of NI-VISA, and then add the file to the project. The visa32.bas module contains all VISA functions and constant statements.



Then, add the Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long) statement into the visa32.bas module; or you can also create a new module to declare the Sleep function.

3. Add the **Label**, **Text**, and **Button** controls. The layout interface for adding controls is as follows:



4. Encapsulate the read and write operations of VISA.
   1) Encapsulate the write operation of VISA for easier operation.

```
'-------------------------------------------------------
'Function Name：InstrWrite
```

```
'Function：Send command to the instrument
'Input：rsrcName,instrument(resource) name
        strCmd,Command
'-------------------------------------------------------
Public Sub InstrWrite(rsrcName As String, strCmd As String)
   Dim status As Long
   Dim dfltRM As Long
   Dim sesn As Long
   Dim rSize As Long

   'Initialize the system
   status = viOpenDefaultRM(dfltRM)
   'Failed to initialize the system
   If (status < VI_SUCCESS) Then
      MsgBox " No VISA resource was opened！ "
      Exit Sub
   End If
   'Open the VISA instrument
   status = viOpen(dfltRM, rsrcName, VI_NULL, VI_NULL, sesn)
   'Failed to open the instrument
   If (status < VI_SUCCESS) Then
      MsgBox "Failed to open the instrument！ "
      Exit Sub
   End If

   'Write command to the instrument
   status = viWrite(sesn, strCmd, Len(strCmd), rSize)
   'Failed to write to the instrument
   If (status < VI_SUCCESS) Then
      MsgBox " Faild to write to the instrument！ "
      Exit Sub
   End If

   'Close the system
   status = viClose(sesn)
   status = viClose(dfltRM)

End Sub
```

2) Encapsulate the read operation of VISA for easier operation.

```
'-------------------------------------------------------
'Function Name：InstrRead
'Function：Read the return value from the instrument
'Input：rsrcName,Resource name
'Return：The string gotten from the instrument
'-------------------------------------------------------
Public Function InstrRead(rsrcName As String) As String
   Dim status As Long
   Dim dfltRM As Long
```

```
    Dim sesn As Long
    Dim strTemp0 As String * 256
    Dim strTemp1 As String
    Dim rSize As Long

    'Begin by initializing the system
    status = viOpenDefaultRM(dfltRM)
    'Initialization failed
    If (status < VI_SUCCESS) Then
        MsgBox " Failed to open the instrument! "
        Exit Function
    End If
    'Open the instrument
    status = viOpen(dfltRM, rsrcName, VI_NULL, VI_NULL, sesn)
    'Failed to open the instrument
    If (status < VI_SUCCESS) Then
        MsgBox " Failed to open the instrument! "
        Exit Function
    End If

    'Read from the instrument
    stasus = viRead(sesn, strTemp0, 256, rSize)
    'Reading failed
    If (status < VI_SUCCESS) Then
        MsgBox " Failed to read from the instrument! "
        Exit Function
    End If

    'Close the system
    status = viClose(sesn)
    status = viClose(dfltRM)

    'Remove the space at the end of the string
    strTemp1 = Left(strTemp0, rSize)
    InstrRead = strTemp1
End Function
```

5. Add the control event codes.

   1) Connect to the instrument

```
    'Connect to the instrument
    Private Sub CmdConnect_Click()
        Const MAX_CNT = 200
        Dim status As Long
        Dim dfltRM As Long
        Dim sesn As Long
        Dim fList As Long
        Dim buffer As String * MAX_CNT, Desc As String * 256
        Dim nList As Long, retCount As Long
        Dim rsrcName(19) As String * VI_FIND_BUFLEN, instrDesc As String *
VI_FIND_BUFLEN
```

```
    Dim i, j As Long
    Dim strRet As String
    Dim bFindRSA As Boolean

    'Initialize the system
    status = viOpenDefaultRM(dfltRM)
    'Initialization failed
    If (status < VI_SUCCESS) Then
        MsgBox " No VISA resource was opened  ！ "
        Exit Sub
    End If

    'Find instrument resource
    Call viFindRsrc(dfltRM, "USB?*INSTR", fList, nList, rsrcName(0))
    'Get the list of the instruments (resources)
    strRet = ""
    bFindRSA = False
    For i = 0 To nList - 1
    'Get the instrument name
    InstrWrite rsrcName(i), "*IDN?"
    Sleep 200
    strRet = InstrRead(rsrcName(i))
    'Continuing searching for the resource until an RSA instrument is found
    strRet = UCase(strRet)
    j = InStr(strRet, "RSA")
    If (j >= 0) Then
       bFindRSA = True
       Exit For
    End If

    Call viFindNext(fList + i - 1, rsrcName(i))
    Next i
    'Display
    If (bFindRSA = True) Then
       TxtInsAddr.Text = rsrcName(i)
    Else
       TxtInsAddr.Text = ""
    End If
End Sub
```

2) Write Operation

```
'Write the command to the instrument
Private Sub CmdWrite_Click()
    If (TxtInsAddr.Text = "") Then
        MsgBox ("Please write the instrument address！ ")
    End If

    InstrWrite TxtInsAddr.Text, TxtCommand.Text
End Sub
```

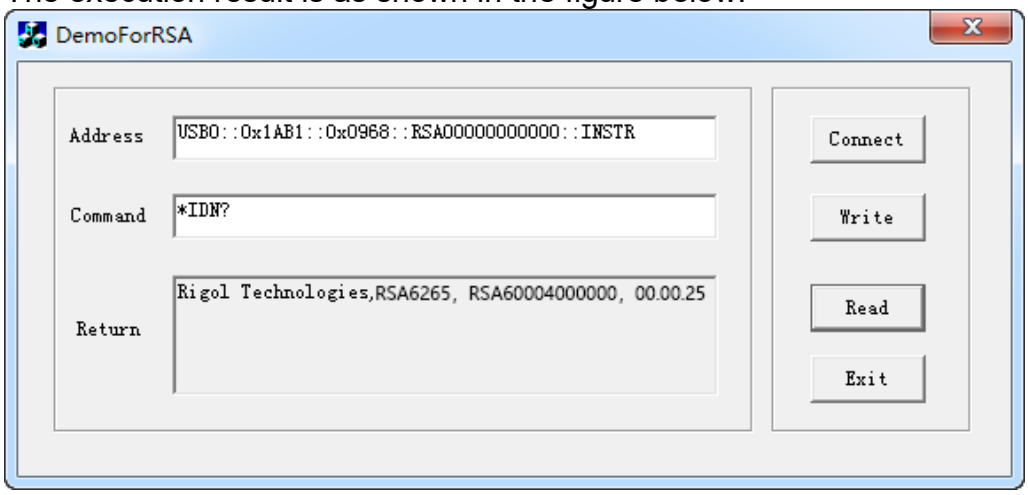
3) Read Operation

```
'Read the return value from the instrument
Private Sub CmdRead_Click()
   Dim strTemp As String
   strTemp = InstrRead(TxtInsAddr.Text)
   TxtReturn.Text = strTemp
End Sub
```

6. Run the results.

1) Click **Connect** to search for the spectrum analyzer;
2) Input "*IDN?" in the "Command" edit box;
3) Click **Write** to write the command into the spectrum analyzer;
4) Click **Read** to read the return value.

The execution result is as shown in the figure below.

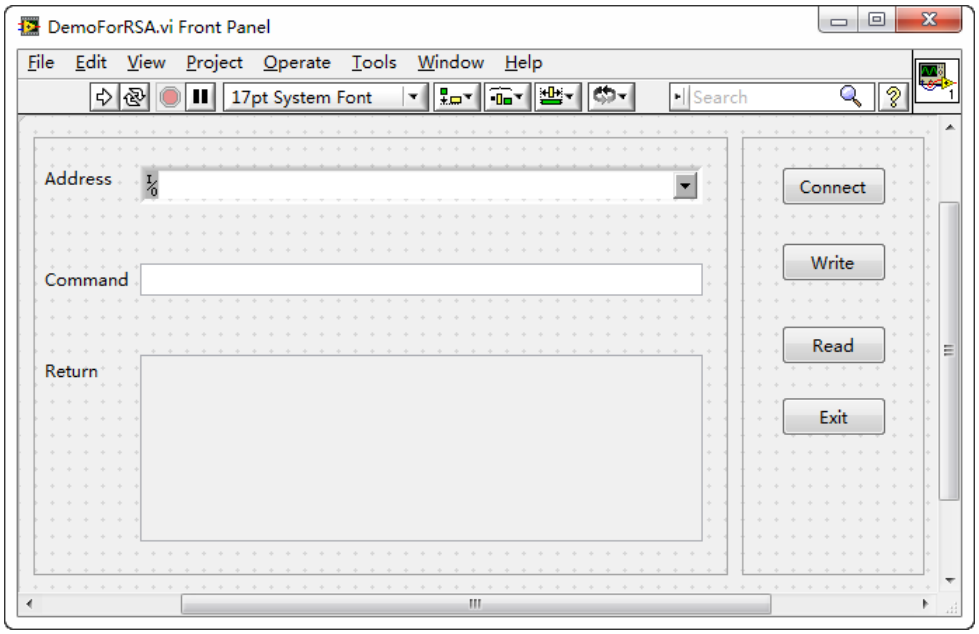


LabVIEW 2010 Programming Example

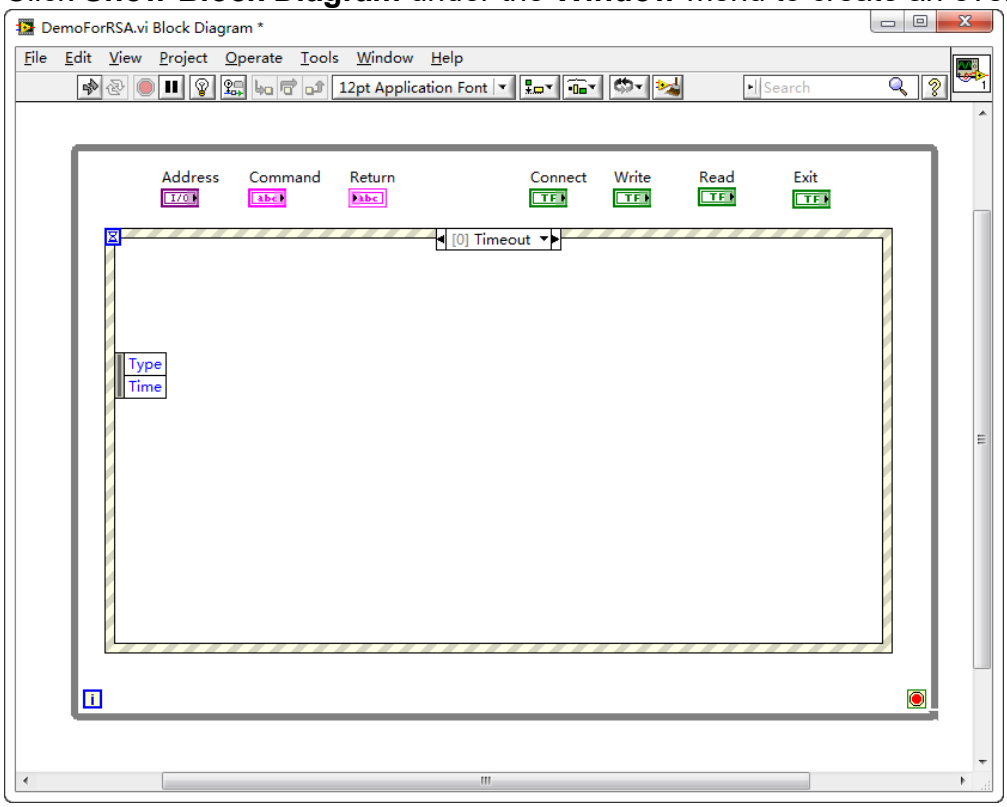
Enter the Labview 2010 programming environment, and perform the following procedures.

1. Create a VI file, and name it "DemoForRSA".

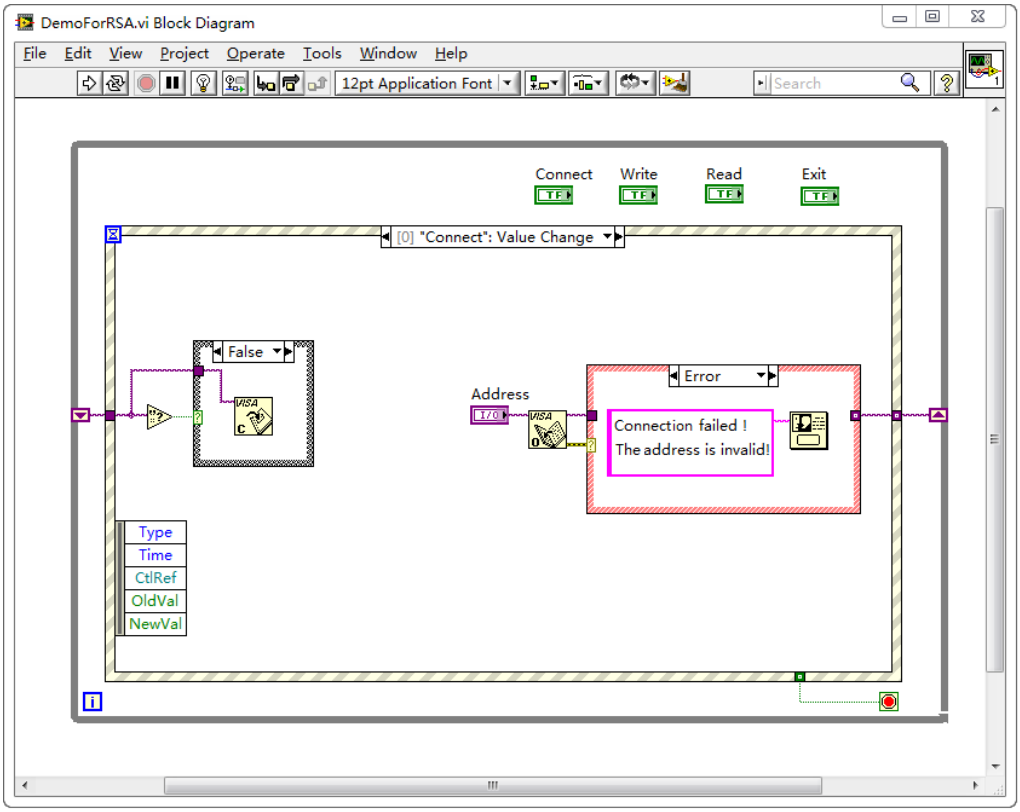
2. Add controls to the front panel interface, including the Address field, Command field, and Return field, the Connect button, the Write button, the Read button, and the Exit button.



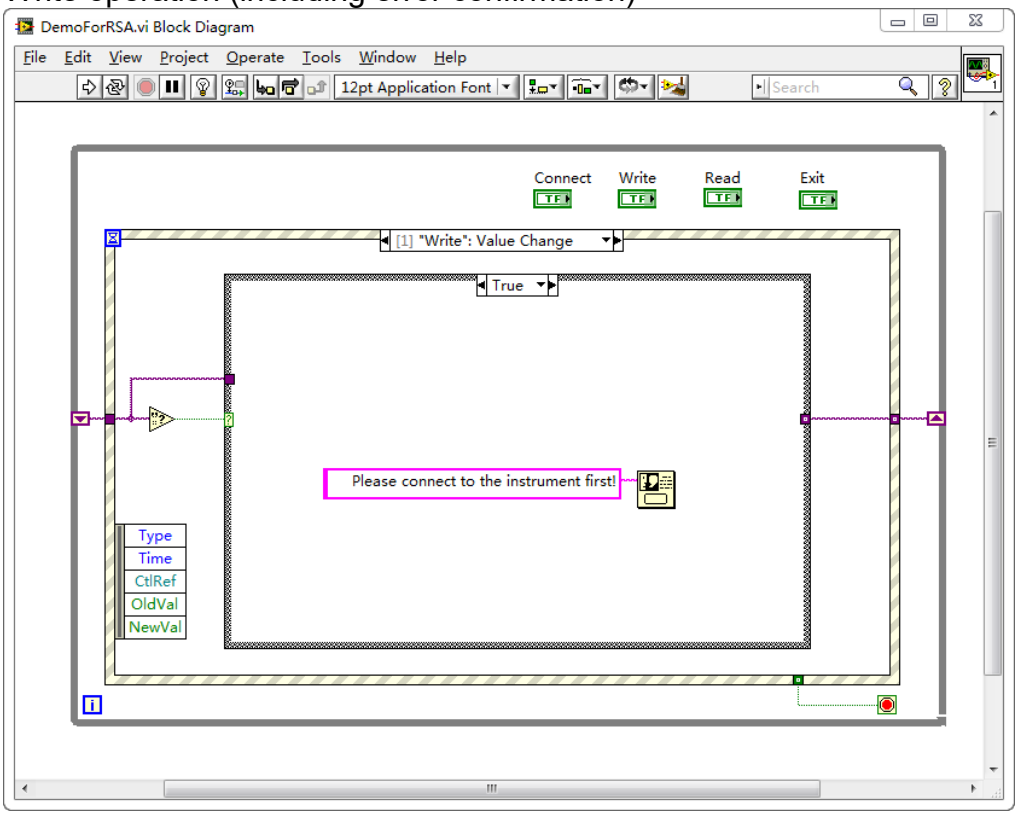
3. Click **Show Block Diagram** under the **Window** menu to create an event structure.

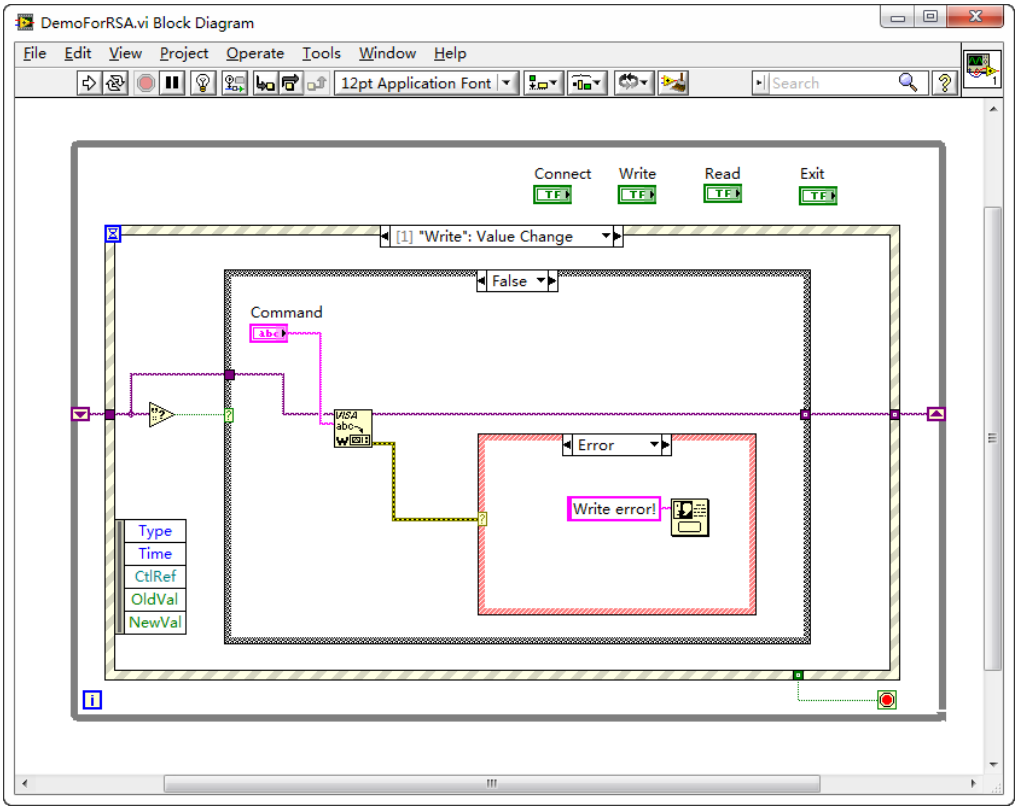


4. Add the events (including connecting to the instrument, write operation, read operation, and exit)
   1) Connect to the instrument

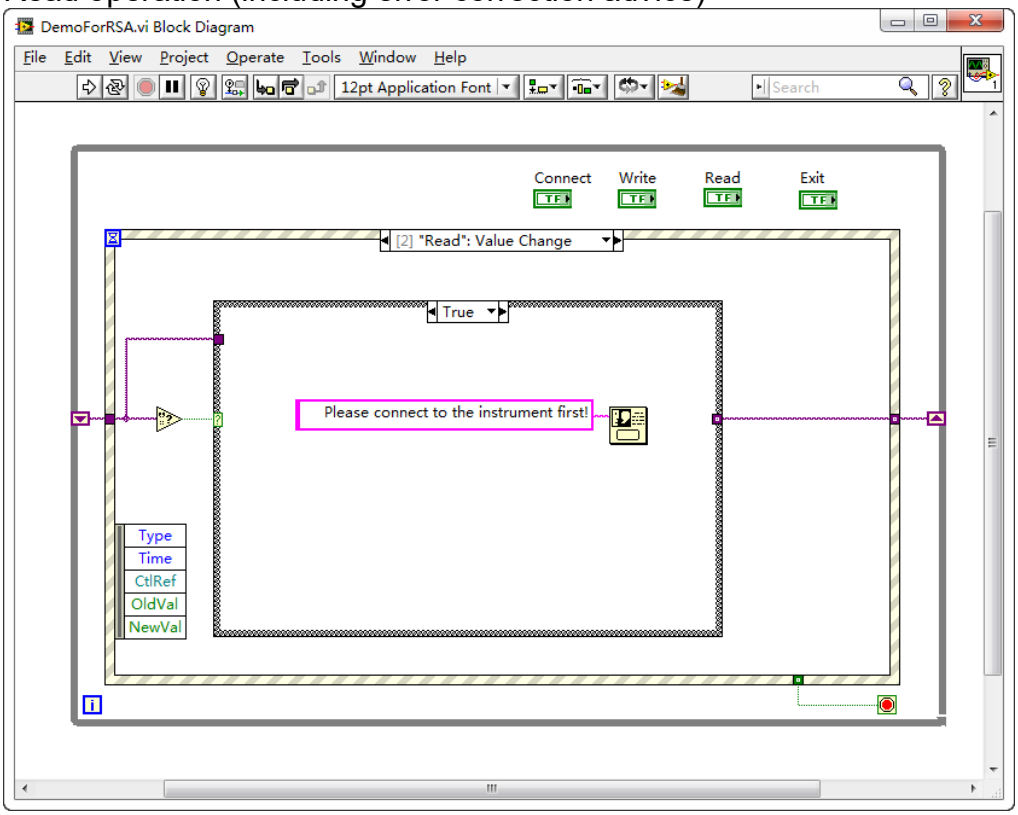


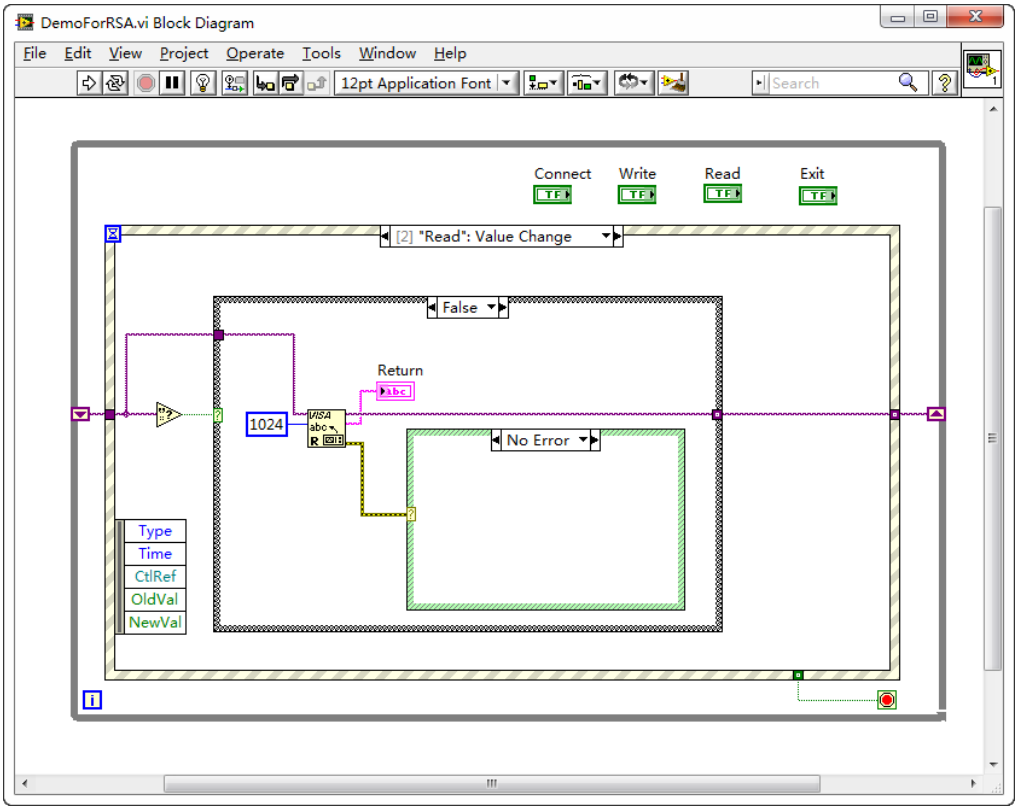
   2) Write operation (including error confirmation)



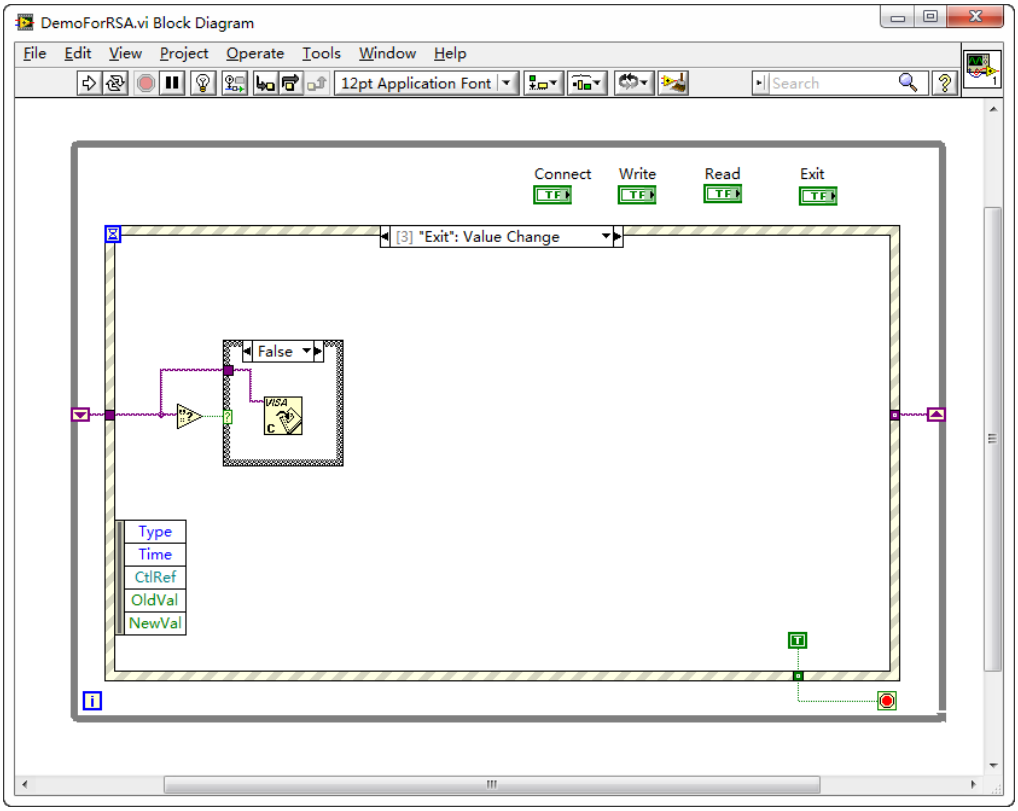


3) Read operation (including error correction advice)



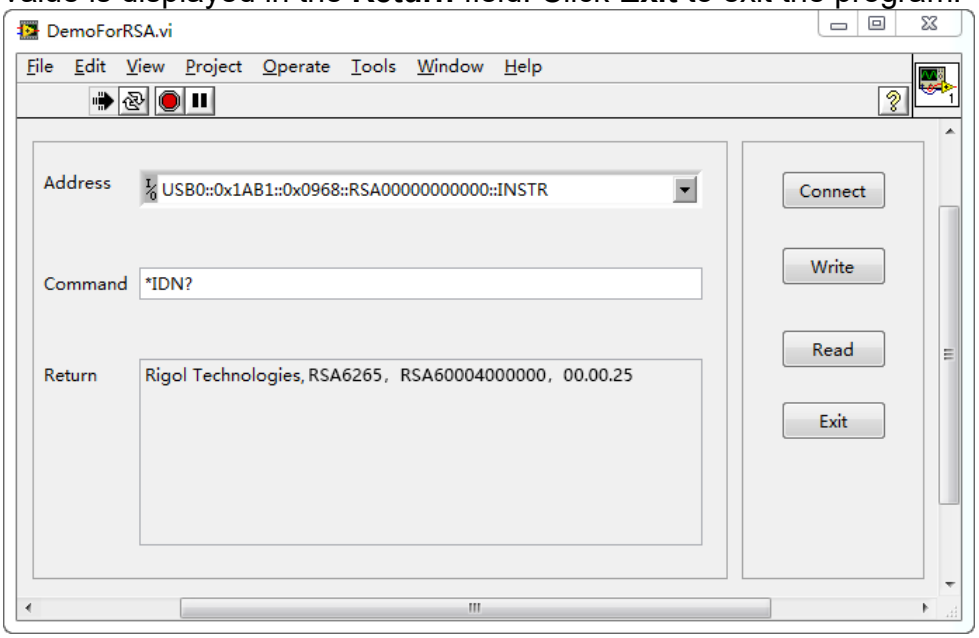


4) Exit



5) Run the program, and then the following interface is displayed below. Click the VISA resource name from the drop-down list under **Address**, and click **Connect** to connect the instrument. Then, input a command in the **Command** field. Click **Write** to write the command to the instrument. If the command is a query (e.g.*IDN?), click

Write to write the command into the instrument, and then click **Read**. The return value is displayed in the **Return** field. Click **Exit** to exit the program.



Linux Programming Example

This section illustrates how to program and control the spectrum analyzer to realize the common functions in Linux operating system.

Programming Preparations

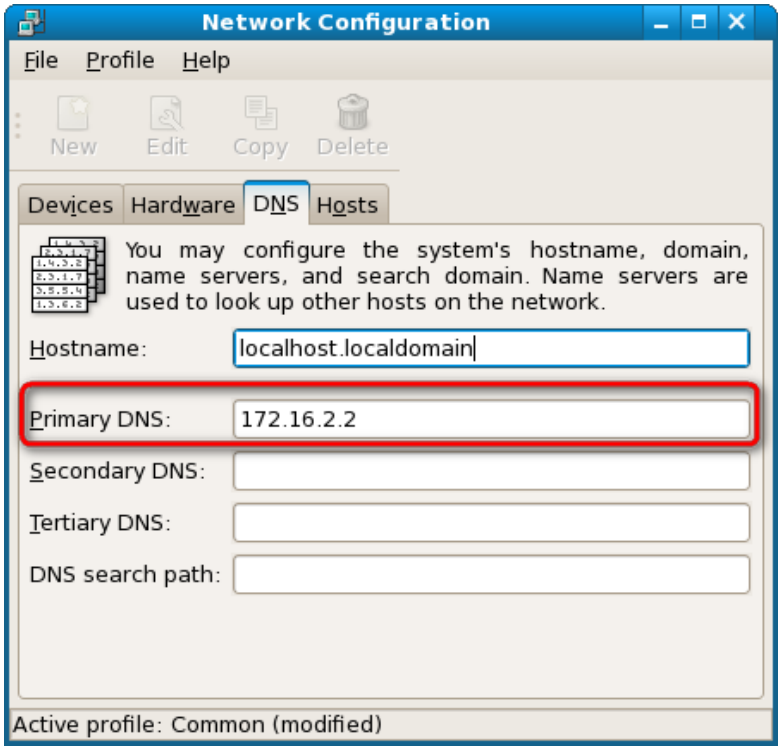
1. Programming environment:
   Operating system: Fedroa 8 (Linux-2.6.23)
   GCC version: gcc-4.1.2

2. Install the VISA library. First, check whether your PC has installed NI's VISA library. If not, download it from NI website (*http://www.ni.com/visa/*). The installation procedures are as follows:
   Download the VISA library NI-VISA-4.4.0.ISO from the NI website.

   Create a new directory.
   #mkdir NI_VISA
   Mount the iso file.
   #mount -o loop -t iso9660 NI-VISA-4.4.0.iso NI_VISA
   Enter the NI_VISA directory to install
   #cd NI_VISA
   #./INSTALL
   Unmount the iso file.
   #umount NI_VISA
   After the installation is finished, the default installation path is /usr/local.

3. Connect spectrum analyzer to the PC via the LAN interface of the analyzer. Use the USB cable to connect the analyzer to the PC via the LAN interface on the rear panel of the analyzer. You can also use a network cable to connect the spectrum analyzer to the local area network where the PC resides.

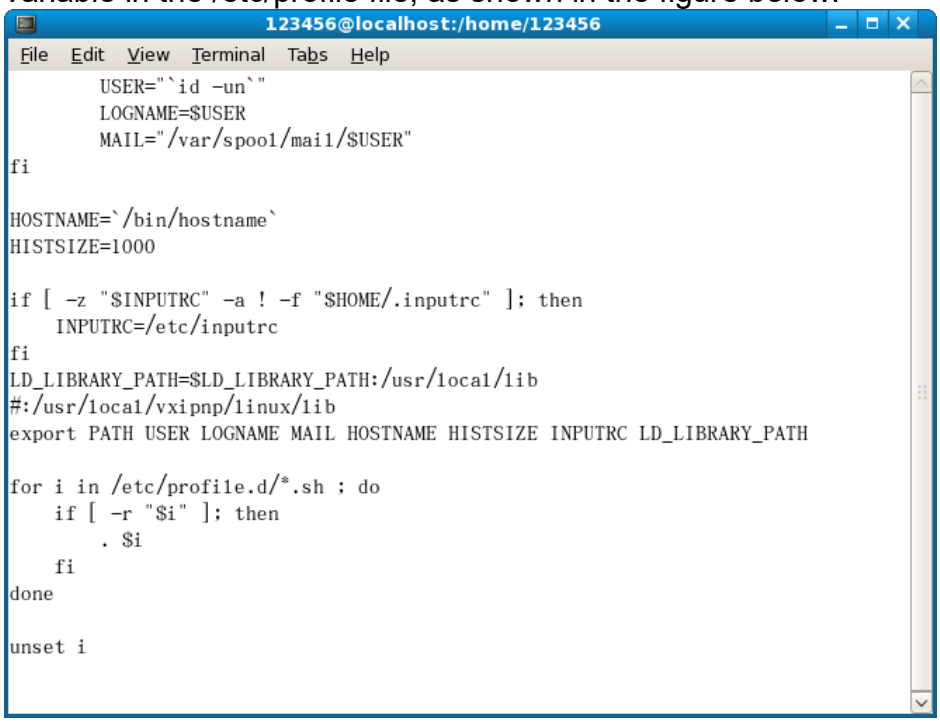
   After the spectrum analyzer is connected to the PC properly, configure the network address for the spectrum analyzer to make its address to be within the same network segment where the PC resides. For example, if the network address and DNS setting configured for the PC are as shown in the figures below, then, the network address of the spectrum analyzer should be configured as follows.
   IP Address: 172.16.3.X*
   Default Gateway: 172.16.3.1
   Subnet Mask: 255.255.255.0
   DNS: 172.16.2.2
   Note*: X can be any value not used ranging from 2 to 254.



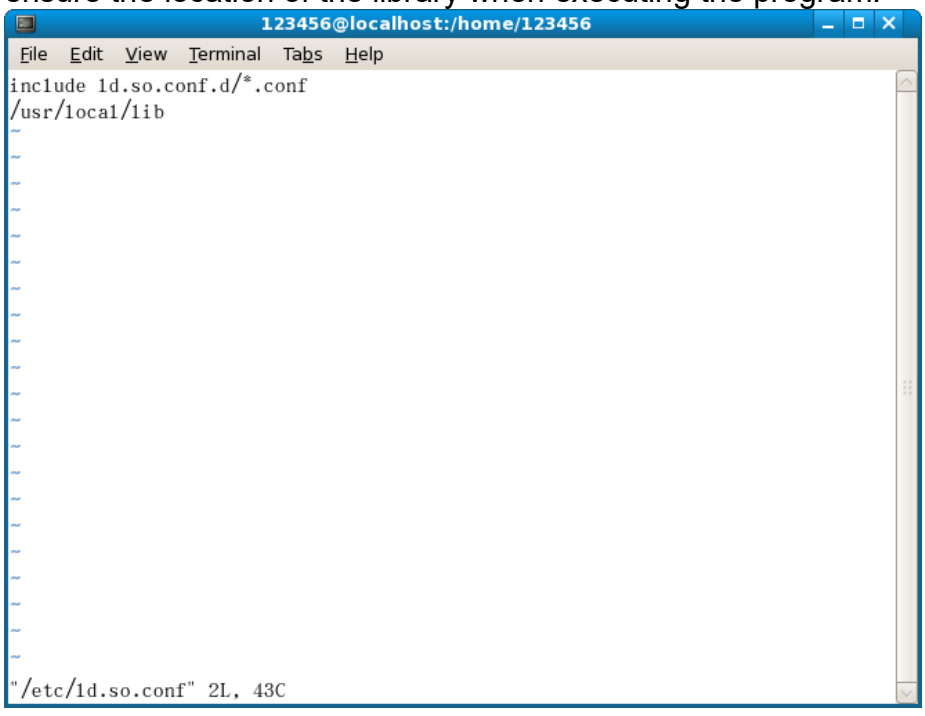
4. Use either of the following two methods to add the library location to the search path of the library, so that the program can load the installed library file automatically.

   Method 1: Specify the search path of the library in the environment variable LD_LIBRARY_PATH.

Operation Method: Add the library file path /usr/local/lib to the LD_LIBRARY_PATH variable in the /etc/profile file, as shown in the figure below.



Method 2: Add the search path of the library in the /etc/ld.so.conf file.
Operation Method: #echo "/usr/local/lib" >> /etc/ld.so.conf, as shown in the figure below.
After setting the search path of the library in /etc/ld.so.conf, run the /sbin/ldconfig command to update /etc/ld.so.cache (this command should have the root permission) to ensure the location of the library when executing the program.



Linux Programming Procedures

1. Edit the DemoForDSA.h header file and declare a class to encapsulate the operation and property of the instrument.

```
#ifndef DEMO_FOR_RSA_H
#define DEMO_FOR_RSA_H

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <iostream>
//#include <syswait.h>
using namespace std;

#define MAX_SEND_BUF_SIZE 50
#define MAX_REC_SIZE 300

class DemoForRSA
{
// Construction
public:
DemoForRSA();
bool InstrRead(string strAddr, string & pstrResult);
bool InstrWrite(string strAddr, string strContent);
bool ConnectInstr();

string m_strInstrAddr;
string m_strResult;
string m_strCommand;

};

void makeupper(string & instr);

#endif
```

2. Edit the DemoForDSA.cpp file to realize various operations of the instrument.

```
#include "visa.h"
#include "DemoForRSA.h"

DemoForRSA::DemoForRSA()
{
m_strInstrAddr = "";
m_strResult = "";
m_strCommand = "";
}

bool DemoForRSA::ConnectInstr()

{
```

```
ViUInt32 retCount;
ViStatus status;
ViSession defaultRM;
ViString expr = "?*";
ViPFindList findList = new unsigned long;
ViPUInt32 retcnt = new unsigned long;
string strSrc = "";
string strInstr = "";
ViChar instrDesc[1000];

unsigned long i = 0;
bool bFindRSA = false;
memset(instrDesc,0,1000);

//Turn on the VISA device
status = viOpenDefaultRM(&defaultRM);

if (status < VI_SUCCESS)
{
   cout<<"No VISA equipment!"<<endl;
   return false;
}

//Search for resources
status = viFindRsrc(defaultRM,expr,findList, retcnt, instrDesc);

for (i = 0;i < (*retcnt);i++)
{
   //Acquire the instrument name
   strSrc = instrDesc;

   InstrWrite(strSrc,"*IDN?");
   usleep(200);
   InstrRead(strSrc,strInstr);

   // If the RSA series is found, then exit
   makeupper(strInstr);
   if (strInstr.find("RSA",0) > 0)
   {
      bFindRSA = true;
      m_strInstrAddr = strSrc;
      break;
    }

   //Acquire the next device
   status = viFindNext(*findList,instrDesc);
}

if (bFindRSA == false)
{
```

```
        printf("RSA device not found!\n");
        return false;
    }

        return true;
    }

    bool DemoForRSA::InstrWrite(string strAddr, string strContent) //Write operation
    {
    ViSession defaultRM,instr;
    ViStatus status;
    ViUInt32 retCount;
    char * SendBuf = NULL;
    char * SendAddr = NULL;
    bool bWriteOK = false;
    string str;
    //Address conversion, convert the string type to char*
    SendAddr = const_cast<char*>(strAddr.c_str());

    //Address conversion, convert the string type to char*
    SendBuf = const_cast<char*>(strContent.c_str());

    //Turn on the specified device□
    status = viOpenDefaultRM(&defaultRM);
    if (status < VI_SUCCESS)
    {
       cout<<"No VISA equipment!"<<endl;
       return false;
    }

    status = viOpen(defaultRM, SendAddr, VI_NULL, VI_NULL, &instr);

    //Write command to the device
    status = viWrite(instr, (unsigned char *)SendBuf, strlen(SendBuf), &retCount);

    //Turn off the device□
    status = viClose(instr);
    status = viClose(defaultRM);
    return bWriteOK;
    }

    bool DemoForRSA::InstrRead(string strAddr, string & pstrResult) //Read operation
    {
    ViSession defaultRM,instr;
    ViStatus status;
    ViUInt32 retCount;
    char* SendAddr = NULL;
    char * result = NULL;
    bool bReadOK = false;
    unsigned char RecBuf[MAX_REC_SIZE];
```

```
string str;
memset(RecBuf,0,MAX_REC_SIZE);

result=char*)malloc(MAX_REC_SIZE*sizeof(char));
memset(result,0,MAX_REC_SIZE);

//Address conversion, convert the string type to char*
SendAddr=const_cast<char*>(strAddr.c_str());

//Turn on the VISA device
status = viOpenDefaultRM(&defaultRM);
if (status < VI_SUCCESS)
{
    // Error Initializing VISA...exiting
    cout<<"No VISA equipment!"<<endl;
    return false;
}

//Turn on the specified device□
status = viOpen(defaultRM, SendAddr, VI_NULL, VI_NULL, &instr);

//Read from the device□
status = viRead(instr, RecBuf, MAX_REC_SIZE, &retCount);

//Turn off the device□
status = viClose(instr);
status = viClose(defaultRM);
sprintf(result,"%s",RecBuf);
pstrResult = result;
free(result);
return bReadOK;
}

void makeupper( string &instr)
{

   string outstr = "";
   if(instr == "")
   {
      exit(0);
   }

   for(int i = 0;i < instr.length();i++)
   {
      instr[i] = toupper(instr[i]);
   }

}
```

3. Edit the function file mainloop.cpp to complete the flow control.

```
#include "DemoForRSA.h"

void menudisplay()
{
   cout<<"\t\t   Please operate the instrument:\n read write quit"<<endl;
}

int main()
{
   DemoForRSA demo;
   char temp[50];
   if(!demo.ConnectInstr())
   {
      cout<<"can not connect the equipment!"<<endl;
      return 0;
    }
   else
   {

      cout<<"\n connect equipment success!"<<endl;
      cout<<" the equipment address is :"<<demo.m_strInstrAddr<<endl;
    }

   while(1)
   {
      menudisplay();
      //cin>>demo.m_strCommand;
      cin.getline(temp,50);
      demo.m_strCommand=temp;
      if(demo.m_strCommand[0]='r' && demo.m_strCommand[1]='e'
         && demo.m_strCommand[2]='a' && demo.m_strCommand[3]='d')
      {
         //demo.InstrWrite(demo.m_strInstrAddr,"*IDN?");
         //demo.InstrRead(demo.m_strInstrAddr,demo.m_strResult);
         cout<<"read result:"<<demo.m_strResult<<endl;
         demo.m_strResult="";

       }

       else if (demo.m_strCommand[0]='w' && demo.m_strCommand[1]='r'
                 && demo.m_strCommand[2]='i' && demo.m_strCommand[3]='t' &&
                demo.m_strCommand[4]='e')
       {
                if (demo.m_strInstrAddr="")
                {
                    cout<<"Please connect the instrument！ \ n";
                 }
                demo.InstrWrite(demo.m_strInstrAddr,demo.m_strCommand.substr(5,40));
                usleep(200);
```

```
                //Read operation
                demo.InstrRead(demo.m_strInstrAddr,demo.m_strResult);

            }

            else if (demo.m_strCommand[0] == 'q' && demo.m_strCommand[1] == 'u'
                    && demo.m_strCommand[2] == 'i' && demo.m_strCommand[3] == 't')

            {
                    break;
            }
            else if(demo.m_strCommand != "")
            {
                    cout<<"Bad command!"<<endl;
            }
        }
        return 1;

    }
```

4. makefile file

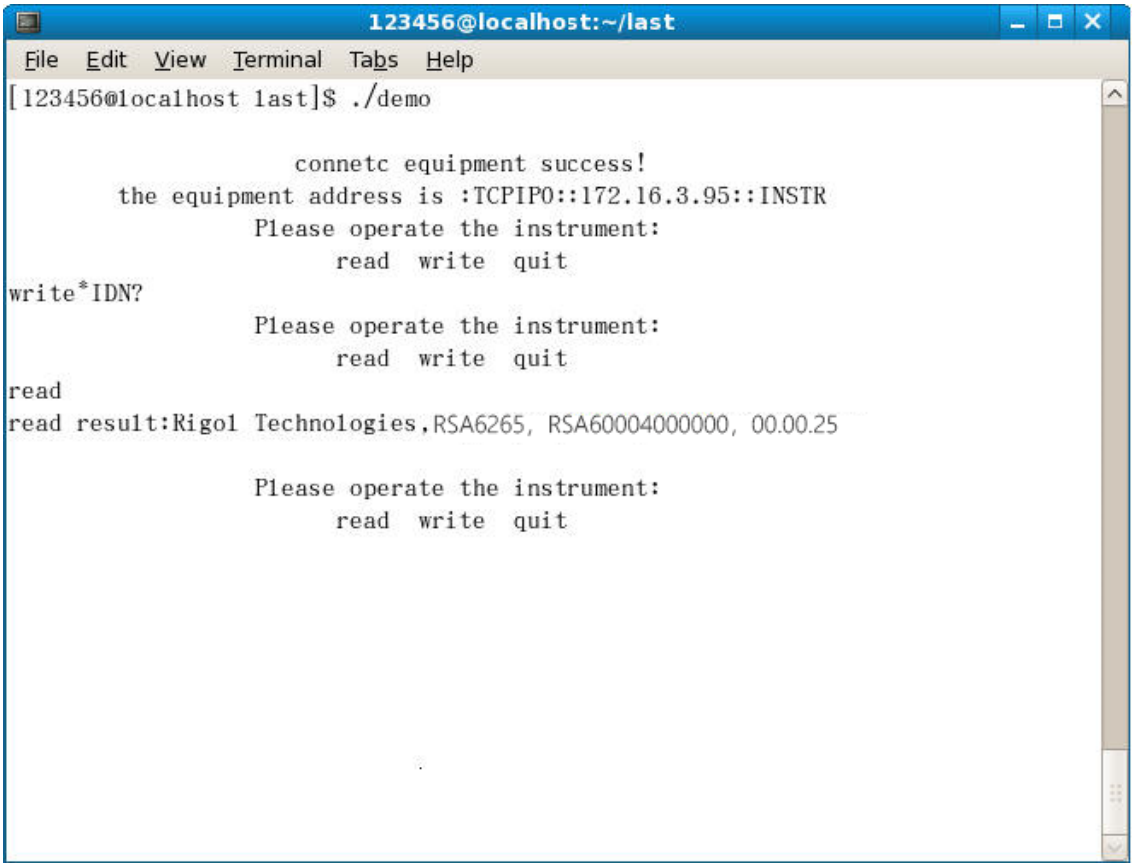
```
src = DemoForRSA.cpp mainloop.cpp DemoForRSA.h

obj = DemoForRSA.o mainloop.o
INCLUDE= -I/usr/local/vxipnp/linux/include
LIB= -lvisa -lc -lpthread
CC=
demo : $(obj)
$(CC) $(INCLUDE) $(LIB) -o demo $(obj)

mainloop.o : mainloop.cpp DemoForRSA.h
$(CC) -c $< -o $@
DemoForRSA.o: DemoForRSA.cpp DemoForRSA.h
$(CC) -c $< -o $@

.PHONY : clean
clean:
rm demo $(obj)
```

5. Run the results.
   1) #make
   2) ./demo
   3) When the program runs, the instrument is connected automatically. If no instrument is found, a prompt message "No VISA equipment!" is displayed, and the system exits the program. If the instrument is found and successfully connected, the following interface is displayed.
   4) Input write<command> (for example, write<*IDN?>) to write the command into the spectrum analyzer.
   5) Input read to read the return value, as shown in the figure below.



Boost Smart World and Technology Innovation

- Cellular-5G/WIFI
- UWB/RFID/ ZIGBEE
- Digital Bus/Ethernet
- Optical Communication

- Digital/Analog/RF Chip
- Memory and MCU Chip
- Third-Generation Semiconductor
- Solar Photovoltaic Cells

- New Energy Automobile
- PV/Inverter
- Power Test
- Automotive Electronics

Provide Testing and Measuring Products and Solutions for Industry Customers

HEADQUARTER

RIGOL TECHNOLOGIES CO., LTD.
No.8 Keling Road, New District,
Suzhou,JiangSu, P.R.China
Tel: +86-400620002
Email: info-cn@rigol.com

EUROPE

RIGOL TECHNOLOGIES EU GmbH
Friedrichshafener Str. 5
82205 Gilching
Germany
Tel: +49(0)8105-27292-21
Email: info-europe@rigol.com

NORTH AMERICA

RIGOL TECHNOLOGIES, USA INC.
10220 SW Nimbus Ave.
Suite K-7
Portland, OR 97223
Tel: +1-877-4-**RIGOL**-1
Email: sales@rigol.com

JAPAN

RIGOL JAPAN CO., LTD.
5F,3-45-6,Minamiotsuka, Toshima-Ku,
Tokyo,170-0005,Japan
Tel: +81-3-6262-8932
Fax: +81-3-6262-8933
Email: info.jp@rigol.com

KOREA

RIGOL KOREA CO,. LTD.
5F, 222, Gonghang-daero,
Gangseo-gu, Seoul, Republic of Korea
Tel: +82-2-6953-4466
Fax: +82-2-6953-4422
Email: info.kr@rigol.com

For Assistance in Other Countries

Email: info.int@rigol.com