



SSG5080A Series Signal Generator

Programming Guide

PG0805A-E01A

| | |
|--|-----------|
| 1. Programming Overview | 5 |
| 1.1 Build Communication..... | 5 |
| 1.1.1 Build Communication Using VISA..... | 5 |
| 1.1.2 Build Communication Using Sockets | 8 |
| 1.1.3 Connecting the signal generator via the USB Host port..... | 8 |
| 1.2 Remote Control Capabilities | 9 |
| 1.2.1 User-defined Programming..... | 9 |
| 1.2.2 Send SCPI Commands via NI-MAX..... | 9 |
| 2. SCPI Overview..... | 14 |
| 2.1 Command Format..... | 14 |
| 2.2 Symbol Instruction..... | 14 |
| 2.3 Parameter Type..... | 15 |
| 2.4 Command Abbreviation..... | 16 |
| 3. SCPI Commands..... | 17 |
| 3.1 IEEE Common Commands..... | 17 |
| 3.1.1 Identification Query (*IDN) | 17 |
| 3.1.2 Reset (*RST)..... | 17 |
| 3.1.3 Clear Status (*CLS)..... | 17 |
| 3.1.4 Standard Event Status Enable (*ESE)..... | 18 |
| 3.1.5 Standard Event Status Register Query (*ESR) | 18 |
| 3.1.6 Operation Complete Query (*OPC) | 18 |
| 3.1.7 Service Request Enable (*SRE)..... | 18 |
| 3.1.8 Status Byte Query (*STB)..... | 19 |
| 3.1.9 Wait-to-Continue (*WAI)..... | 19 |
| 3.1.10 Self Test Query (*TST) | 19 |
| 3.2 System Subsystem..... | 19 |
| 3.2.1 System Time (:SYSTem:TIME) | 19 |
| 3.2.2 System Date (:SYSTem:DATE) | 20 |
| 3.2.3 IP Address (:SYSTem:COMMunicate:LAN:IPAddress)..... | 20 |
| 3.2.4 Gateway (:SYSTem:COMMunicate:LAN:GATeway)..... | 20 |
| 3.2.5 Subnet Mask (:SYSTem:COMMunicate:LAN:SMASK) | 21 |
| 3.2.6 IP Config (:SYSTem:COMMunicate:LAN:TYPE)..... | 21 |
| 3.2.7 Language (SYSTem:LANGuage) | 21 |
| 3.2.8 Screen Saver (SYSTem:SCReen:SAVer) | 22 |
| 3.2.9 Beeper (SYSTem:ALARm) | 22 |
| 3.2.10 Setup Type (:SYSTem:PON:TYPE)..... | 22 |
| 3.2.11 Power On Line (SYSTem:POWeron:TYPE)..... | 23 |
| 3.2.12 IOM Adjustment State (:SYSTem:REF:DAC:STAT) | 23 |
| 3.2.13 Ref Osc Code (:SYSTem:REF:DAC) | 23 |
| 3.2.14 Ref Osc Code Store (:SYSTem:REF:DAC:SAVE)..... | 24 |
| 3.2.15 Ref Osc Code Load (:SYSTem:REF:DAC:LOAD)..... | 24 |
| 3.2.16 Reset Ref Osc Code to Default (:SYSTem:REF:DAC:DEFault) | 24 |
| 3.2.17 GPIB Address (SYSTem:GPIB) | 25 |

| | | |
|------------|--|-----------|
| 3.3 | Preset Subsystem | 25 |
| 3.3.1 | Preset (:SOURce:PRESet) | 25 |
| 3.3.2 | System Preset (:SYSTem:PRESet) | 25 |
| 3.3.3 | Preset Save (:SYSTem:PRESet:SAVE) | 26 |
| 3.3.4 | Preset Path (:SYSTem:PRESet:PATH) | 26 |
| 3.3.5 | Preset Type (:SYSTem:PRESet:TYPE) | 26 |
| 3.3.6 | Factory Reset (:SYSTem:FDEFault) | 27 |
| 3.3.7 | Reset & Clear (SYSTem:RESet:CLEar) | 27 |
| 3.4 | Output Subsystem | 27 |
| 3.4.1 | RF Output (:OUTPut[:STATe]) | 27 |
| 3.4.2 | RF Output ([:SOURce]:OUTPut) | 28 |
| 3.5 | Source Subsystem | 28 |
| 3.5.1 | [:SOURce]:FREQuency Subsystem..... | 28 |
| 3.5.2 | [:SOURce]:POWer Subsystem..... | 30 |
| 3.5.3 | [:SOURce]:SWEep Subsystem..... | 40 |
| 3.5.4 | [:SOURce]:MODulation Subsystem..... | 49 |
| 3.5.5 | [:SOURce]:AM Subsystem | 50 |
| 3.5.6 | [:SOURce]:FM Subsystem | 52 |
| 3.5.7 | [:SOURce]:PM Subsystem | 57 |
| 3.5.8 | [:SOURce]:PULM Subsystem..... | 59 |
| 3.5.9 | [:SOURce]:LFOutput Subsystem | 68 |
| 3.5.10 | [:SOURce]:LFOutput:SWEep Subsystem | 71 |
| 3.6 | Sense Subsystem..... | 74 |
| 3.6.1 | Sensor Info (:SENSe[:POWer]:TYPE?) | 74 |
| 3.6.2 | Sensor State (:SENSe[:POWer]:STATus) | 75 |
| 3.6.3 | Measurement (:SENSe[:POWer]:VALue?) | 75 |
| 3.6.4 | Statistics State (:SENSe[:POWer]:STATISTics:STATe) | 75 |
| 3.6.5 | Statistics Value (:READ[:POWer]?) | 76 |
| 3.6.6 | Statistics Max Value (:SENSe[:POWer]:STATISTics:MAX?) | 76 |
| 3.6.7 | Statistics Min Value (:SENSe[:POWer]:STATISTics:MIN?) | 76 |
| 3.6.8 | Statistics Mean Value (:SENSe[:POWer]:STATISTics:AVG?) | 77 |
| 3.6.9 | Statistics Count (:SENSe[:POWer]:STATISTics:COUNt?) | 77 |
| 3.6.10 | Statistics Clear (:SENSe[:POWer]:STATISTics:CLEAr) | 77 |
| 3.6.11 | Auto Zero (:CALibration:ZERO:TYPE) | 78 |
| 3.6.12 | Zeroing (:SENSe[:POWer]:ZERO)..... | 78 |
| 3.6.13 | Frequency Type (:SENSe[:POWer]:SOURce) | 78 |
| 3.6.14 | Frequency (:SENSe[:POWer]:FREQuency) | 79 |
| 3.6.15 | Level Offset State (:SENSe[:POWer]:OFFSet:STATe) | 79 |
| 3.6.16 | Level Offset (:SENSe[:POWer]:OFFSet) | 79 |
| 3.6.17 | Average Type (:SENSe[:POWer]:FILTer:TYPE) | 80 |
| 3.6.18 | Average Times (:SENSe[:POWer]:FILTer:LENGth) | 80 |
| 3.6.19 | Internal Noise (:SENSe[:POWer]:FILTer:NSRatio) | 80 |
| 3.6.20 | Logging (:SENSe[:POWer]:LOGGing:STATe) | 81 |
| 4. | Programming Examples | 81 |
| 4.1 | VISA Examples..... | 81 |
| 4.1.1 | VC++ Example..... | 81 |
| 4.1.2 | VB Example | 86 |
| 4.1.3 | MATLAB Example | 90 |

| | | |
|------------|------------------------------|-----------|
| 4.1.4 | LabVIEW Example | 92 |
| 4.2 | Socket Examples | 94 |
| 4.2.1 | Python Example | 94 |
| 4.2.2 | Telnet Example | 96 |

1. Programming Overview

The SSG5080A supports both USB and LAN interfaces. By using these interfaces, in combination with NI-VISA and programming languages, users can remotely control the signal generator. The instrument comes with an embedded web interface; VXI-11, Sockets and Telnet protocols can be used to communicate with the signal generator. This chapter introduces how to build communication between the signal generator and the PC. It also introduces the remote control capabilities.

1.1 Build Communication

1.1.1 Build Communication Using VISA

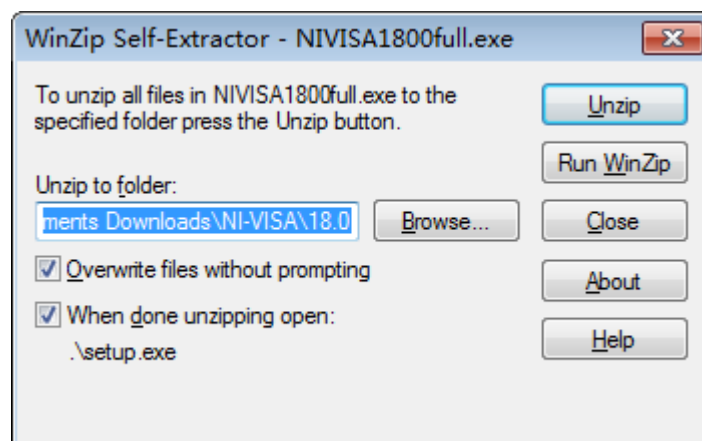
1. Install NI-VISA

Before programming, you will need to install NI-VISA, which you can download from the National Instruments VISA web site. There are full and Run-Time Engine versions of NI-VISA. The full version includes the NI device driver and a tool named NI MAX which is a user interface to control the device. The Run-Time Engine version is a smaller file than the full version only includes the NI device driver.

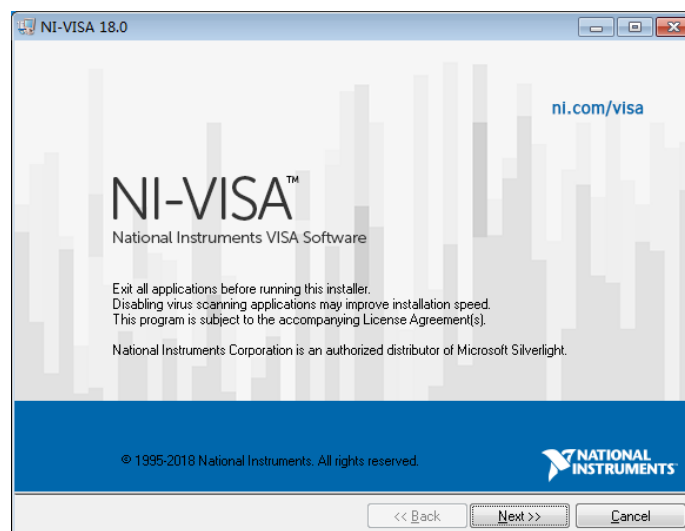
For example, you can get NI-VISA 18.0 full version from:
<http://www.ni.com/download/ni-visa-18.0/7597/en/>.

You can also download NI-VISA Run-Time Engine 18.0 to your PC and install it as default selection. Its installation process is similar with the full version. After you downloaded the file you can follow the steps below to install it:

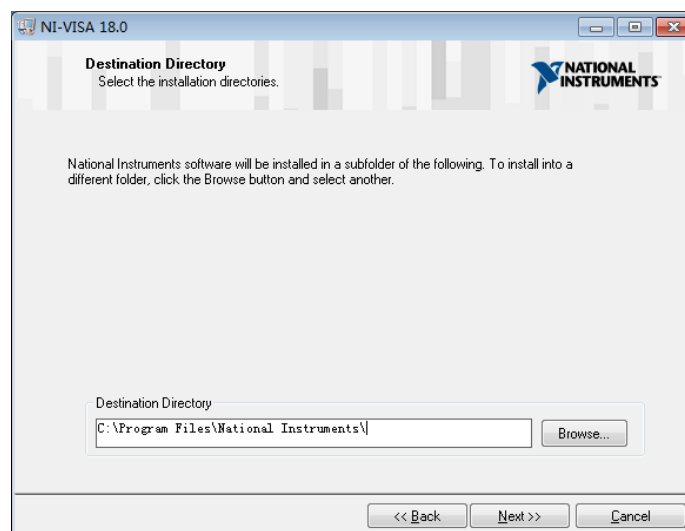
a. Double click the NIVISA1800full.exe, dialog shown as below:



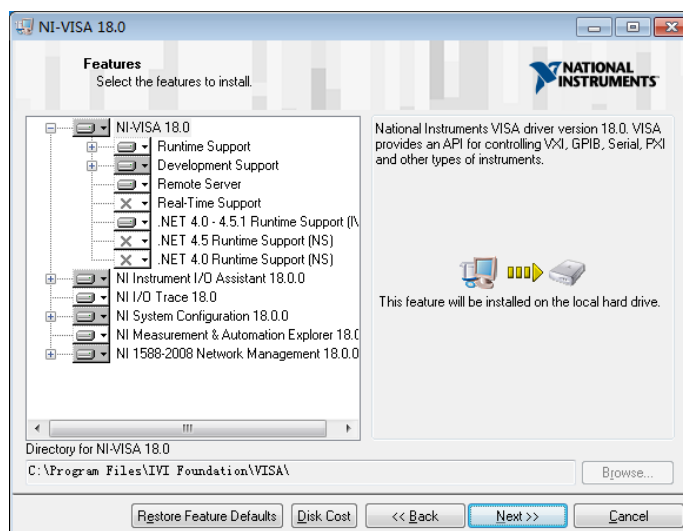
b. Click Unzip, the installation process will automatically launch after unzipping files. If your computer needs to install .NET Framework 4.6.2, its setup process will auto start.



c. The NI-VISA installing dialog is shown above. Click Next to start the installation process.



Set the install path, default path is “C:\Program Files\National Instruments\”, you can change it. Click Next, dialog shown as above.



d. Click Next twice, in the License Agreement dialog, select the “I accept the above 2 License Agreement(s).”, and click Next, “Start Installation” dialog shown.

e. Click Next to run installation.

f.

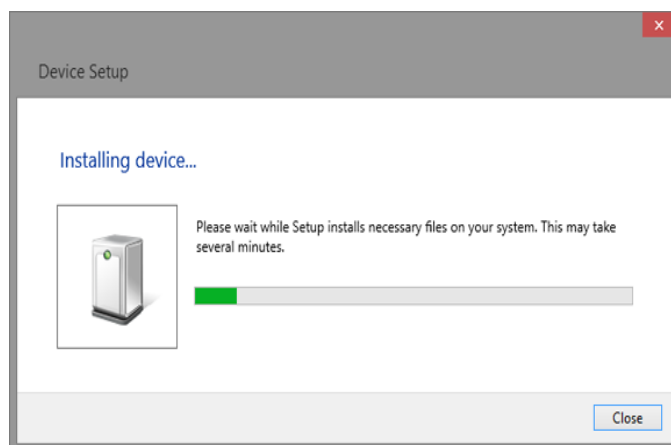


Now the installation is complete, reboot your PC.

2. Connect the Instrument

Depending on your specific model, your signal generator may be able to communicate with a PC through the USB or LAN interface. This manual uses the USB connection in the examples. (For instructions to communicate with a PC through the LAN interface see the User Manual.)

a. Connect the USB Device interface at the rear panel of the signal generator and the USB Host interface of the PC using a USB cable. Assuming your PC is already turned on, turn on your signal generator and your PC will display the “Device Setup” screen as it automatically installs the device driver as shown below.



b. Wait for the installation to complete and then proceed to the next step.

1.1.2 Build Communication Using Sockets

LAN communication using Sockets uses the Transmission Control Protocol/Internet Protocol (TCP/IP) layer that is included with many operating systems. A socket is a fundamental technology used for computer networking and allows applications to communicate using standard mechanisms built into network hardware and operating systems. The method accesses a port on the signal generator from which bidirectional communication with a network computer can be established. Unlike VISA, this technique uses currently available resources and doesn't require additional software/hardware to run.

Before you can use sockets, you must select the signal generator socket port number to use:

- Standard mode. Available on port 5025. Use this port for simple programming.
- Telnet mode. The telnet SCPI service is available on port 5024.

1.1.3 Connecting the signal generator via the USB Host port

Refer to the following steps to finish the connection via USB:

1. Install NI-VISA on your PC for GPIB driver.
2. Connect the signal generator USB Host port to a PC's GPIB card port, with SIGLENT USB-GPIB adaptor.



3. Switch on the signal generator.
4. Press button on the front panel **System** → Interface → GPIB to enter the

GPIO number.

The signal generator will be detected automatically as a new GPIO point.

1.2 Remote Control Capabilities

1.2.1 User-defined Programming

Users can use SCPI commands to program and control the signal generator. For details, refer to the introductions in “Programming Examples”.

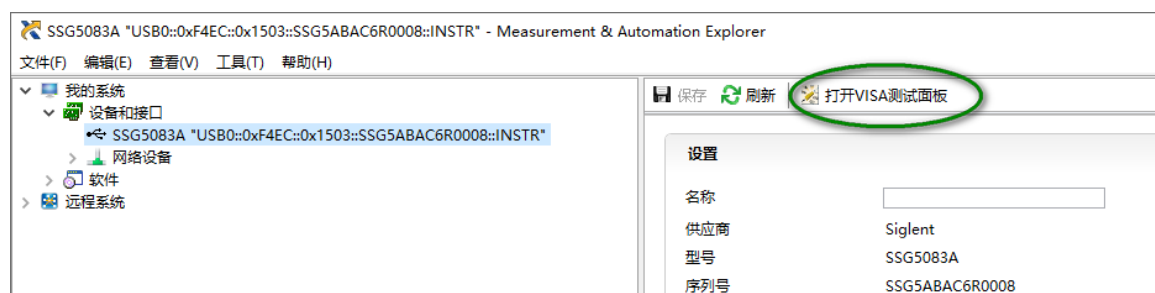
1.2.2 Send SCPI Commands via NI-MAX

Users can control the signal generator remotely by sending SCPI commands via NI-MAX software.

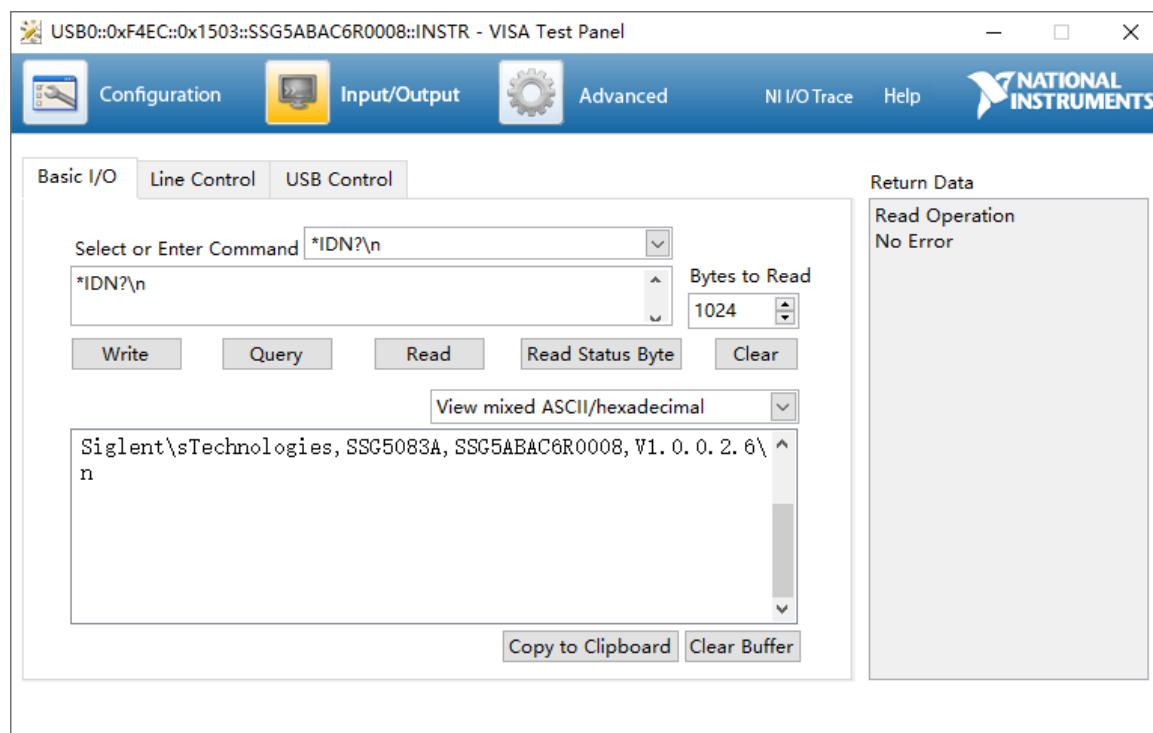
1.2.2.1 Using USB

Run NI MAX software.

1. Click “Device and interface” at the upper left corner of the software.
2. Find the “USBTC” device symbol.



3. Click “Open VISA Test Panel” option button, then the following interface will appear.
4. Click the “Input/Output” option button and click the “Query” option button in order to view the operation information.



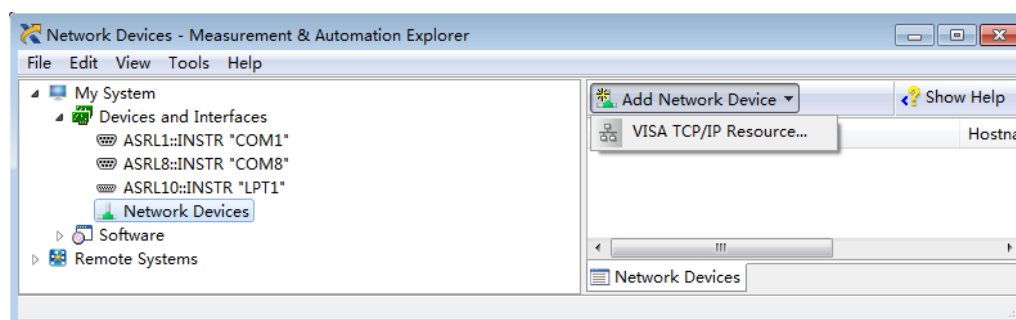
NOTE: The “*IDN?” command (known as the Identification Query) returns the instrument manufacturer, instrument model, serial number, and other identification information.

1.2.2.2 Using LAN

Add a Network Device, and select a VISA TCP/IP Resource as shown:

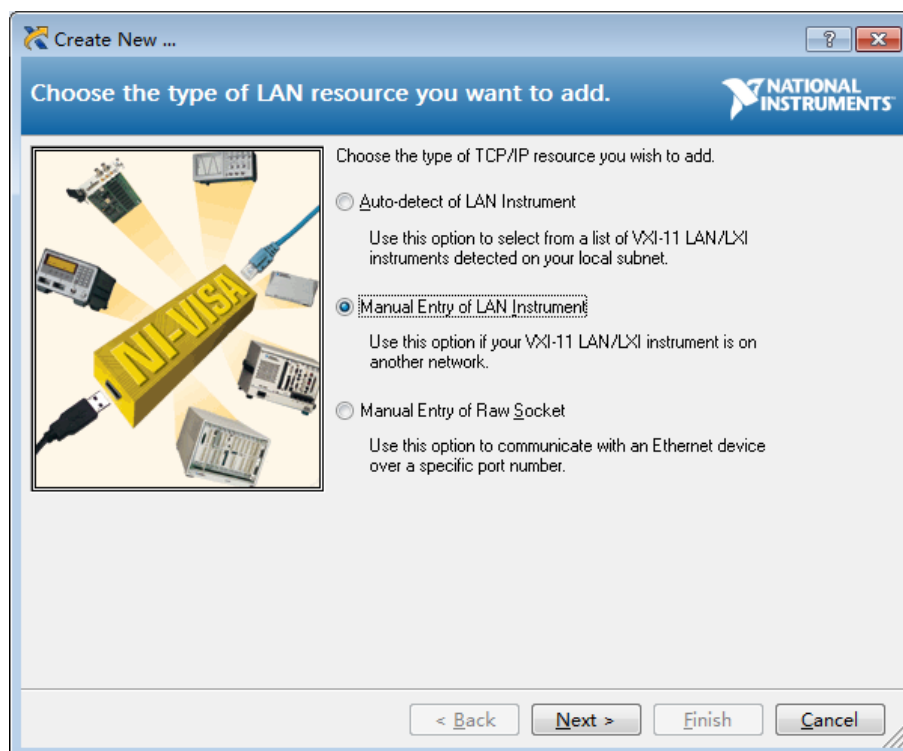
Run NI MAX software.

1. Click “Device and interface” at the upper left corner of the software
2. Find the “Network Devices” symbol, click “Add Network Devices”

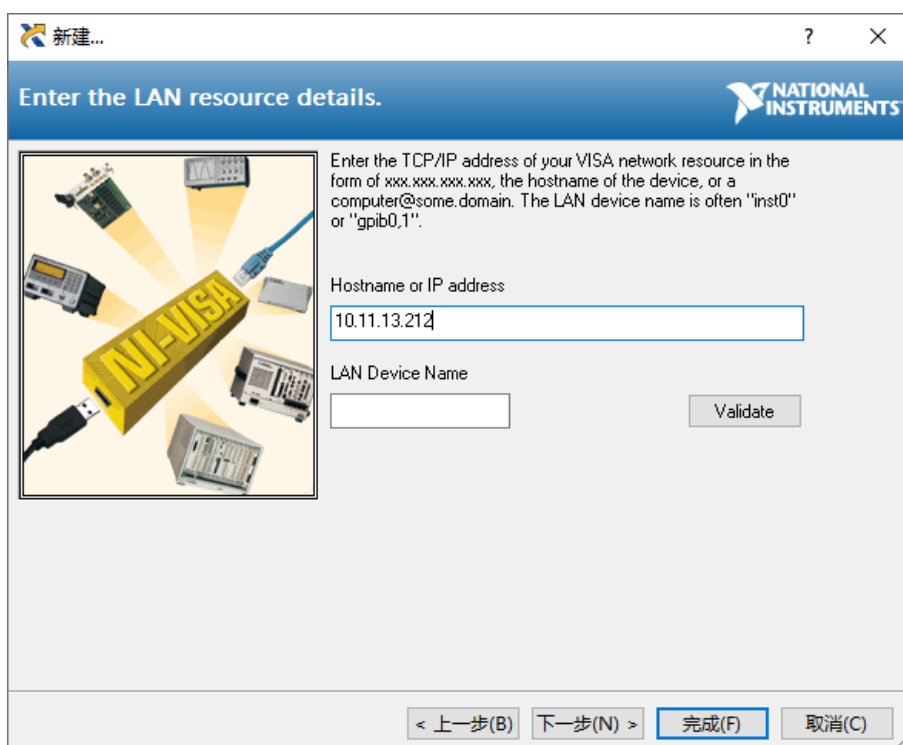


3. Select Manual Entry of LAN instrument, select Next, and enter the IP address as shown.

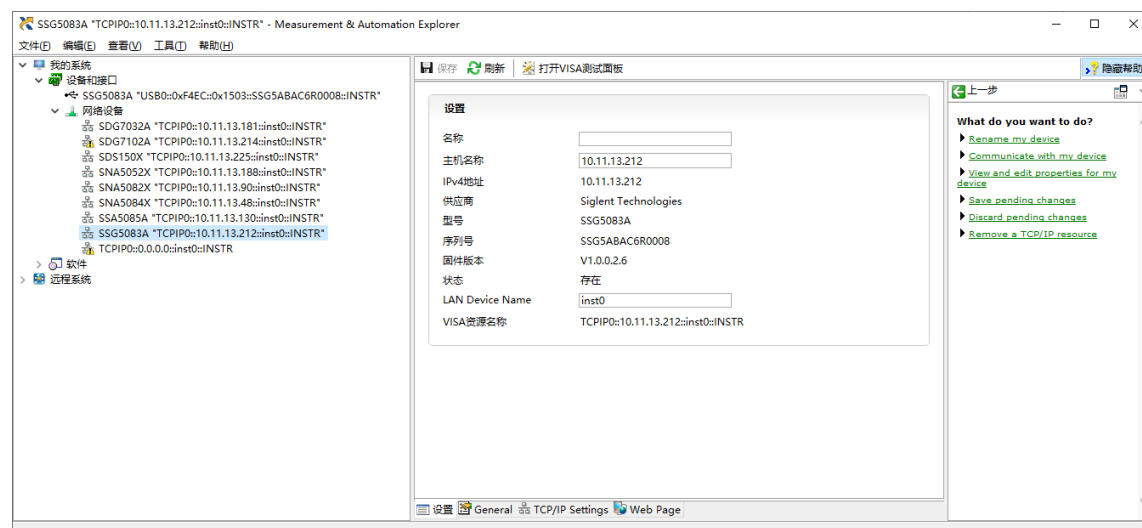
Click Finish to establish the connection:



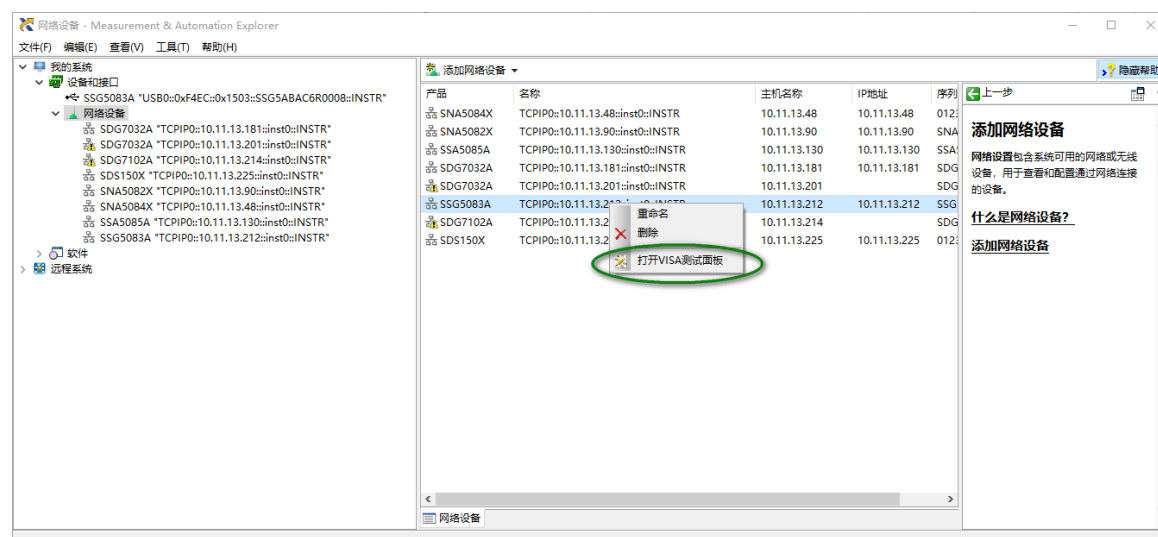
NOTE: Leave the LAN Device Name BLANK or the connection will fail.



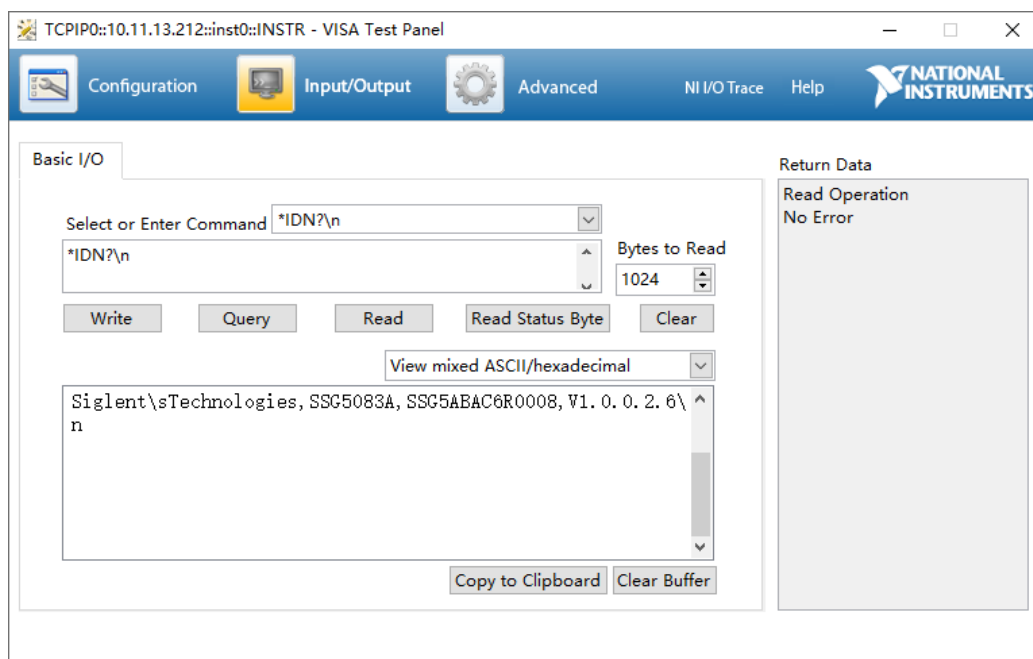
4. After a brief scan, the connection should be shown under Network Devices:



5. Right-click on the product and select Open NI-VISA Test Panel:



6. Click "Input/Output" option button and click "Query" option button. If everything is OK, you will see the Read operation information returned as shown below.



2. SCPI Overview

2.1 Command Format

SCPI commands present a hierarchical tree structure containing multiple subsystems; each of the subsystems is made up of a root keyword and several sub keywords. The command string usually starts with a colon “:”, the keywords are separated by a colon “:” and the parameter settings are separated by spaces. Query commands add a question mark “?” to the end of the string.

For example:

```
:SOURce:FREQuency <freq>
```

```
:SOURce:FREQuency?
```

SOURce is the root key of the command, FREQuency is second.

The command begins with “:”, and separates the keywords at the same time, <freq> separated by space and represents the parameter available for setting; “?” represents a query. A query sent to the instrument indicates that the instrument will have a response string. Therefore, queries ask a question and expect a response.

2.2 Symbol Instruction

The following four symbols are not the content of SCPI commands and cannot be sent with the commands, but are used to describe certain aspects of the commands.

1. Triangle Brackets < >

The parameter in the triangle brackets must be replaced by an effective value. For example:

Send the “POWer:SPC:TARGet <power>” command in “POWer:SPC:TARGet 0”.

2. Square Brackets []

The content in the square brackets can be ignored. When the parameter is ignored, the instrument will set the parameter to its default.

For example,

In the “[:SOURce]:POWer?” command, sending either of the commands below can generate the same effect:

```
:SOURce:POWer?
```

```
:POWer?
```

3. Vertical Bar |

The vertical bar is used to separate multiple parameters and when sending the command, you can choose one of the parameters.

For example,

In the “[:SOURce]:AM:STATe OFF|ON|0|1” command, the parameters available are “OFF”, “ON”, “0” or “1”.

4. Braces { }

The parameters in the braces are optional which can be ignored or set for one or more times.

2.3 Parameter Type

The parameters in the commands introduced in this manual include 6 types: Boolean, enumeration, integer, float and string.

1. Boolean

The parameter in the command could be “OFF”, “ON”, “0” or “1”.

For example:

```
[:SOURce]:FM:STATe OFF|ON|0|1
```

2. Enumeration

The parameter could be any of the values listed.

For example:

```
[:SOURce]:SWEep:STATe OFF|FREQuency|LEVel|LEV_FREQ
```

Valid parameters are “OFF”, “FREQuency”, “LEVel” or LEV_FREQ.

3. Integer

Except other notes, the parameter can be any integer within the effective value range.

For example:

```
[:SOURce]:SWEep:STEP:POINTs <value>
```

The parameter <value> can be set to any integer between 2 and 65535.

4. Float

The parameter can be any value within the effective value range according to the accuracy requirement (the default accuracy contains up to 9 digits after the decimal points).

For example:

```
[:SOURce]:POWer:OFFSet <value>
```

The parameter <value> can be set to any real number between -100 and 100.

5. String

The parameter should be the combinations of ASCII characters.

For example:

```
:SYSTem:COMMunicate:LAN:IPAdDress < "xxx.xxx.xxx.xxx" >
```

The IP address can be set as the string "192.168.1.12".

2.4 Command Abbreviation

All of the commands are not case sensitive, so you can use any of them. But if an abbreviation is used, all the capital letters in the command must be written completely.

For example:

```
:CORRection:FLATness:COUNt?
```

Can be abbreviated to:

```
:CORR:FLAT:COUN?
```


3. SCPI Commands

This chapter introduces the Siglent Technologies SSG5000A SCPI commands, including:

| | |
|----------------------|---------------------|
| IEEE Common Commands | 3.1 |
| System Subsystem | 3.2 |
| Preset Subsystem | 3.3 |
| Output Subsystem | 3.4 |
| Source Subsystem | 3.5 |
| Sense Subsystem | 3.6 |

3.1 IEEE Common Commands

3.1.1 Identification Query (*IDN)

| | |
|----------------|---|
| Command Format | *IDN? |
| Instruction | Returns an instrument identification information string. The string contains the manufacturer, model number, serial number, software number, FPGA number and CPLD number. |
| Menu | None |
| Example | *IDN? Return: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6 |

3.1.2 Reset (*RST)

| | |
|----------------|--|
| Command Format | *RST |
| Instruction | This command presets the instrument to a factory defined condition that is appropriate for remote programming operation. *RST is equivalent to performing the two commands :SOURce:PRESet and *CLS. This command always performs a factory preset. |
| Menu | None |
| Example | *RST |

3.1.3 Clear Status (*CLS)

| | |
|----------------|--|
| Command Format | *CLS |
| Instruction | Clears the status byte register. It does this by emptying the error queue and clearing all bits in all of the event registers. The status byte register summarizes the states of the other registers. It is also |

| | |
|---------|--|
| | responsible for generating service requests. |
| Menu | None |
| Example | *CLS |

3.1.4 Standard Event Status Enable (*ESE)

| | |
|-------------------|---|
| Command Format | *ESE <number> *ESE? |
| Instruction | Set the bits in the standard event status enable register. This register monitors I/O errors and synchronization conditions such as operation complete, request control, query error, device dependent error, execution error, command error and power on. A summary bit is generated on execution of the command. The query returns the state of the standard event status enable register. |
| Menu | None |
| Example | *ESE 16 |

3.1.5 Standard Event Status Register Query (*ESR)

| | |
|-------------------|---|
| Command Format | *ESR? |
| Instruction | Queries and clears the standard event status event register. (This is a destructive read.) The value returned reflects the current state (0/1) of all the bits in the register. |
| Menu | None |
| Example | *ESR? |

3.1.6 Operation Complete Query (*OPC)

| | |
|-------------------|---|
| Command Format | *OPC *OPC? |
| Instruction | Set bit 0 in the standard event status register to “1” when all pending operations have finished. The query stops any new commands from being processed until the current processing is complete. Then it returns a “1”, and the program continues. This query can be used to synchronize events of other instruments on the external bus. Returns a “1” if the last processing is complete. Use this query when there’s a need to monitor the command execution status, such as a sweep execution. |
| Menu | None |
| Example | *OPC? |

3.1.7 Service Request Enable (*SRE)

| | |
|-------------------|---|
| Command Format | *SRE <integer> *SRE? |
| Instruction | This command enables the desired bits of the service request enable |

| | |
|---------|--|
| | register. The query returns the value of the register, indicating which bits are currently enabled. |
| Menu | None |
| Example | *SRE 1 |

3.1.8 Status Byte Query (*STB)

| | |
|----------------|---|
| Command Format | *STB |
| Instruction | This query is used by some instruments for a self test. |
| Menu | None |
| Example | *STB |

3.1.9 Wait-to-Continue (*WAI)

| | |
|----------------|---|
| Command Format | *WAI |
| Instruction | This command causes the instrument to wait until all pending commands are completed before executing any additional commands. There is no query form to the command. |
| Menu | None |
| Example | *WAI |

3.1.10 Self Test Query (*TST)

| | |
|----------------|---|
| Command Format | *TST? |
| Instruction | This query is used by some instruments for a self test. |
| Menu | None |
| Example | *TST? |

3.2 System Subsystem

3.2.1 System Time (:SYSTem:TIME)

| | |
|-----------------|---|
| Command Format | :SYSTem:TIME <hhmmss> :SYSTem:TIME? |
| Instruction | Set the System time Get the System time |
| Parameter Type | String |
| Parameter Range | Hours(0 ~ 23), minutes(0 ~ 59), seconds(0 ~ 59) |
| Return | String |
| Default | None |
| Menu | Utility > Setting > Time Setting |
| Example | Set System time: |

| | |
|--|--|
| | :SYSTem:TIME 182559 Get System time: :SYSTem:TIME? |
|--|--|

3.2.2 System Date (:SYSTem:DATE)

| | |
|-----------------|--|
| Command | :SYSTem:DATE <yyyymmdd> |
| Format | :SYSTem:DATE? |
| Instruction | Set system date Get system date |
| Parameter Type | String |
| Parameter Range | Years(four digits), month(1 ~ 12), date(1 ~ 31) |
| Return | String |
| Default | None |
| Menu | Utility > Setting > Time Setting |
| Example | Set System date: :SYSTem:DATE 20050101 Get System date: :SYSTem:DATE? |

3.2.3 IP Address (:SYSTem:COMMunicate:LAN:IPAdDress)

| | |
|-----------------|---|
| Command | :SYSTem:COMMunicate:LAN:IPAdDress < “xxx.xxx.xxx.xxx” > |
| Format | :SYSTem:COMMunicate:LAN:IPAdDress? |
| Instruction | Set the IP address. The IP address will be fetched automatically if the IP assignment is set to DHCP. Get the IP address |
| Parameter Type | String |
| Parameter Range | Conforms to the IP address standard(0-255:0-255:0-255:0-255) |
| Return | IP address string |
| Default | None |
| Menu | Utility > Interface > LAN Setting > IP Address |
| Example | :SYSTem:COMMunicate:LAN:IPAdDress “192.168.1.12” :SYSTem:COMMunicate:LAN:IPAdDress? |

3.2.4 Gateway (:SYSTem:COMMunicate:LAN:GATeway)

| | |
|-------------|---|
| Command | :SYSTem:COMMunicate:LAN:GATeway < “xxx.xxx.xxx.xxx” > |
| Format | :SYSTem:COMMunicate:LAN:GATeway? |
| Instruction | Set the gateway for the signal generator in the network. The gateway will be fetched automatically if the IP assignment is set to DHCP. Get the gateway. |
| Parameter | String |

| | |
|-----------------|---|
| Type | |
| Parameter Range | Conforms to the IP standard (0~255.0~255.0~255.0~255) |
| Return | Gateway string |
| Default | None |
| Menu | Utility > Interface > LAN Setting > Gateway |
| Example | :SYSTem:COMMunicate:LAN:GATeway "192.168.1.1" :SYSTem:COMMunicate:LAN:GATeway? |

3.2.5 Subnet Mask (:SYSTem:COMMunicate:LAN:SMASk)

| | |
|-----------------|---|
| Command Format | :SYSTem:COMMunicate:LAN:SMASk < "xxx.xxx.xxx.xxx" > :SYSTem:COMMunicate:LAN:SMASk? |
| Instruction | Set the subnet mask according to the network settings. The subnet mask will be set automatically if the IP assignment is set to DHCP. Get the subnet mask. |
| Parameter Type | String |
| Parameter Range | Conforms to the IP standard (0-255:0-255:0-255:0-255) |
| Return | Subnet mask string |
| Default | None |
| Menu | Utility > Interface > LAN Setting > Subnet Mask |
| Example | :SYSTem:COMMunicate:LAN:SMASk "255.255.255.0" :SYSTem:COMMunicate:LAN:SMASk? |

3.2.6 IP Config (:SYSTem:COMMunicate:LAN:TYPE)

| | |
|-----------------|--|
| Command Format | :SYSTem:COMMunicate:LAN:TYPE STATIC DHCP :SYSTem:COMMunicate:LAN:TYPE? |
| Instruction | Toggles the IP assignment setting between static (manual) and DHCP (dynamic assignment) mode. Get the IP config mode. |
| Parameter Type | Enumeration |
| Parameter Range | STATIC DHCP |
| Return | Enumeration |
| Default | None |
| Menu | Utility > Interface > LAN Setting > DHCP State |
| Example | :SYSTem:COMMunicate:LAN:TYPE DHCP :SYSTem:COMMunicate:LAN:TYPE? |

3.2.7 Language (SYSTem:LANGuage)

| | |
|----------------|---|
| Command Format | :SYSTem:LANGuage CHINese ENGLish :SYSTem:LANGuage? |
| Instruction | Set language. Get language. |

| | |
|-----------------|---|
| Parameter Type | Enumeration |
| Parameter Range | CHINese ENGLish |
| Return | Enumeration |
| Default | None |
| Menu | Utility > Setting > Language |
| Example | :SYSTem:LANGuage CHINese :SYSTem:LANGuage? |

3.2.8 Screen Saver (SYSTem:SCReen:SAVer)

| | |
|-----------------|---|
| Command Format | SYSTem:SCReen:SAVer OFF 10S 1MIN 5MIN 15MIN 30MIN 1HOUR 2HOUR SYSTem:SCReen:SAVer? |
| Instruction | Set screen saver. Get screen saver. |
| Parameter Type | Enumeration |
| Parameter Range | OFF 10S 1MIN 5MIN 15MIN 30MIN 1HOUR 2HOUR |
| Return | Enumeration |
| Default | OFF |
| Menu | Utility > Setting > Screen Saver |
| Example | SYSTem:SCReen:SAVer 30MIN SYSTem:SCReen:SAVer? |

3.2.9 Beeper (SYSTem:ALARm)

| | |
|-----------------|--|
| Command Format | SYSTem:ALARm ON OFF 1 0 SYSTem:ALARm? |
| Instruction | Set system beeper state. Get system beeper state. |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | ON |
| Menu | Utility > Setting > Beeper |
| Example | SYSTem:ALARm ON SYSTem:ALARm? |

3.2.10 Setup Type (:SYSTem:PON:TYPE)

| | |
|----------------|--|
| Command Format | :SYSTem:PON:TYPE DFT LAST :SYSTem:PON:TYPE? |
| Instruction | Sets the signal generator power on state. Default is the factory configuration and last recalls all of the settings used before the last power down. |

| | |
|-----------------|---|
| | Get power on type. |
| Parameter Type | Enumeration |
| Parameter Range | DFT LAST DFT: Default LAST: Last |
| Return | Enumeration |
| Default | DFT |
| Menu | Utility > Setting > Setup Type |
| Example | :SYSTem:PON:TYPE DFT :SYSTem:PON:TYPE? |

3.2.11 Power On Line (SYSTem:POWeron:TYPE)

| | |
|-----------------|--|
| Command | SYSTem:POWeron:TYPE ON OFF 1 0 |
| Format | SYSTem:POWeron:TYPE? |
| Instruction | Set the signal generator power on line state. Get the signal generator power on line state. |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | OFF |
| Menu | Utility > Setting > Power On Line |
| Example | SYSTem:POWeron:TYPE ON SYSTem:POWeron:TYPE? |

3.2.12 10M Adjustment State (:SYSTem:REF:DAC:STAT)

| | |
|-----------------|--|
| Command | :SYSTem:REF:DAC:STAT ON OFF 1 0 |
| Format | :SYSTem:REF:DAC:STAT? |
| Instruction | Set 10M Adjustment State. Get 10M Adjustment State. |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | Utility > Setting > 10M Adjustment |
| Example | :SYSTem:REF:DAC:STAT ON :SYSTem:REF:DAC:STAT? |

3.2.13 Ref Osc Code (:SYSTem:REF:DAC)

| | |
|-------------|-------------------------|
| Command | :SYSTem:REF:DAC <value> |
| Format | :SYSTem:REF:DAC? |
| Instruction | Set ref osc code. |

| | |
|-----------------|---|
| | Get ref osc code. |
| Parameter Type | Int |
| Parameter Range | 0 ~ 65535 |
| Return | Int |
| Default | 26214 |
| Menu | Utility > Setting > 10M Adjustment > Ref Osc Code |
| Example | :SYSTem:REF:DAC 43000 :SYSTem:REF:DAC? |

3.2.14Ref Osc Code Store (:SYSTem:REF:DAC:SAVE)

| | |
|-----------------|---|
| Command Format | :SYSTem:REF:DAC:SAVE <file_name> |
| Instruction | Save the ref osc code in file. |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | Utility > Setting > 10M Adjustment > Save Ref Osc Setting |
| Example | :SYSTem:REF:DAC:SAVE "U-disk3/test.dac" |

3.2.15Ref Osc Code Load (:SYSTem:REF:DAC:LOAD)

| | |
|-----------------|---|
| Command Format | :SYSTem:REF:DAC:LOAD <file_name> |
| Instruction | Load existing ref osc code files. |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | Utility > Setting > 10M Adjustment > Recall Ref Osc Setting |
| Example | :SYSTem:REF:DAC:LOAD "U-disk3/test.dac" |

3.2.16Reset Ref Osc Code to Default (:SYSTem:REF:DAC:DEFault)

| | |
|----------------|--------------------------------------|
| Command Format | :SYSTem:REF:DAC:DEFault |
| Instruction | Reset ref osc code to default value. |
| Parameter Type | None |
| Parameter | None |

| | |
|---------|---|
| Range | |
| Return | None |
| Default | None |
| Menu | Utility > Setting > 10M Adjustment > Reset to Default |
| Example | :SYSTem:REF:DAC:DEFault |

3.2.17 GPIB Address (SYSTem:GPIB)

| | |
|-----------------|--|
| Command | SYSTem:GPIB <value> |
| Format | SYSTem:GPIB? |
| Instruction | Set GPIB address of the signal source. Get GPIB address of the signal source. |
| Parameter Type | Integer |
| Parameter Range | 1 ~ 30 |
| Return | Integer |
| Default | 18 |
| Menu | Utility > Interface > GPIB Address |
| Example | SYSTem:GPIB 10 SYSTem:GPIB? |

3.3 Preset Subsystem

3.3.1 Preset (:SOURce:PRESet)

| | |
|----------------|--|
| Command | :SOURce:PRESet |
| Format | |
| Instruction | Presets all parameters which are related to the selected signal path |
| Parameter Type | None |
| Return | None |
| Default | None |
| Menu | None |
| Example | SOUR:PRES |

3.3.2 System Preset (:SYSTem:PRESet)

| | |
|----------------|--|
| Command | :SYSTem:PRESet |
| Format | |
| Instruction | According to the preset type, preset the parameter configuration of the machine. |
| Parameter Type | None |
| Return | None |
| Default | None |
| Menu | Utility > Preset |
| Example | For example, preset signal generator to default configuration: |

| | |
|--|--|
| | <code>:SYSTem:PRESet:TYPE DFT</code> <code>:SYSTem:PRES</code> Or preset signal generator to current configuration: <code>:SYSTem:PRESet:TYPE USER</code> <code>:SYSTem:PRESet:SAVE</code> <code>:SYSTem:PRES</code> Or preset signal generator to configuration saved in an existing xml file: <code>:SYSTem:PRESet:TYPE USER</code> <code>:SYSTem:PRESet:PATH "Local/test.xml"</code> <code>:SYSTem:PRES</code> |
|--|--|

3.3.3 Preset Save (:SYSTem:PRESet:SAVE)

| | |
|----------------|---|
| Command Format | <code>:SYSTem:PRESet:SAVE</code> |
| Instruction | Save status for preset when preset type is user |
| Parameter Type | None |
| Return | None |
| Default | None |
| Menu | Utility > Preset |
| Example | <code>:SYSTem:PRESet:SAVE</code> |

3.3.4 Preset Path (:SYSTem:PRESet:PATH)

| | |
|----------------|--|
| Command Format | <code>:SYSTem:PRESet:PATH <path></code> |
| Instruction | Set preset file when preset type is user |
| Parameter Type | String |
| Return | None |
| Default | None |
| Menu | Utility > Setting > Preset Type |
| Example | <code>:SYSTem:PRESet:PATH "Local/test.xml"</code> <code>:SYSTem:PRESet:PATH "U-disk1/test.xml"</code> |

3.3.5 Preset Type (:SYSTem:PRESet:TYPE)

| | |
|-----------------|---|
| Command Format | <code>:SYSTem:PRESet:TYPE DFT USER</code> <code>:SYSTem:PRESet:TYPE?</code> |
| Instruction | Uses this command to preset the signal generator to default or user. Get preset type. |
| Parameter Type | Enumeration |
| Parameter Range | DFT: Default USER: Custom Configuration |

| | |
|---------|---|
| Return | Enumeration |
| Default | DFT |
| Menu | Utility > Setting > Preset Type |
| Example | :SYSTem:PRESet:TYPE DFT :SYSTem:PRESet:TYPE? |

3.3.6 Factory Reset (:SYSTem:FDEFault)

| | |
|-----------------|--|
| Command Format | :SYSTem:FDEFault |
| Instruction | Set both the measure and setting parameters to the factory settings. |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | Utility > Setting > Factory Reset |
| Example | :SYSTem:FDEFault |

3.3.7 Reset & Clear (SYSTem:RESet:CLEar)

| | |
|-----------------|--|
| Command Format | SYSTem:RESet:CLEar |
| Instruction | Set both the measure and setting parameters to the factory settings, and at the same time clear the files saved by the user in the "Local" folder. |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | Utility > Setting > Reset & Clear |
| Example | SYSTem:RESet:CLEar |

3.4 Output Subsystem

3.4.1 RF Output (:OUTPut[:STATe])

| | |
|-----------------|--|
| Command Format | :OUTPut[:STATe] ON OFF 1 0 :OUTPut[:STATe]? |
| Instruction | Activate/Deactivate the RF output Get the state of the RFoutput |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |

| | |
|---------|------------------------|
| Return | Boolean |
| Default | 0 |
| Menu | RF |
| Example | :OUTPut ON :OUTPut? |

3.4.2 RF Output ([:SOURce]:OUTPut)

| | |
|-----------------|-----------------------------------|
| Command Format | [:SOURce]:OUTPut ON OFF 1 0 |
| Instruction | Activate/Deactivate the RF output |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | None |
| Default | 0 |
| Menu | RF |
| Example | SOURce:OUTPut ON |

3.5 Source Subsystem

3.5.1 [:SOURce]:FREQuency Subsystem

3.5.1.1 Frequency Display ([:SOURce]:FREQuency:DISPlay)

| | |
|-----------------|--|
| Command Format | [:SOURce]:FREQuency:DISPlay <freq> [:SOURce]:FREQuency:DISPlay? |
| Instruction | Set the frequency display on parameter bar Get the frequency display on parameter bar |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | Frequency offset + Full frequency range |
| Return | Float, unit: Hz |
| Default | Maximum frequency |
| Menu | Freq |
| Example | FREQuency:DISPlay 2 MHz :FREQuency:DISPlay? |

3.5.1.2 Frequency ([:SOURce]:FREQuency)

| | |
|----------------|--|
| Command Format | [:SOURce]:FREQuency <freq> [:SOURce]:FREQuency? |
| Instruction | Set the frequency of the RF output signal Get the frequency of the RF output signal |

| | |
|-----------------|--|
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | Full frequency range |
| Return | Float, unit: Hz |
| Default | Maximum frequency |
| Menu | FREQ > Frequency |
| Example | FREQuency 2 MHz :FREQuency? |

3.5.1.3 Frequency Offset ([:SOURce] :FREQuency :OFFSet)

| | |
|-----------------|--|
| Command Format | [:SOURce] :FREQuency :OFFSet <freq> [:SOURce] :FREQuency :OFFSet? |
| Instruction | Set the frequency offset of a downstream circuit element Get the frequency offset of a downstream circuit element |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | -200 GHz ~ 200 GHz |
| Return | Float, unit: Hz |
| Default | 0 Hz |
| Menu | FREQ > Freq Offset |
| Example | FREQuency :OFFSet 2 MHz FREQuency :OFFSet? |

3.5.1.4 Phase Offset ([:SOURce] :PHASe)

| | |
|-----------------|--|
| Command Format | [:SOURce] :PHASe <phase> [:SOURce] :PHASe? |
| Instruction | Set the phase of the RF output signal Get the phase of the RF output signal |
| Parameter Type | Float, unit: deg |
| Parameter Range | -360 deg ~ 360 deg |
| Return | Float, unit: deg |
| Default | 0 deg |
| Menu | FREQ > Phase Offset |
| Example | PHASe 20 PHASe? |

3.5.1.5 Phase Reset ([:SOURce] :PHASe :RESet [:SOURce] :PHASe :REF)

| | |
|----------------|--|
| Command Format | [:SOURce] :PHASe :RESet [:SOURce] :PHASe :REF |
|----------------|--|

| | |
|-----------------|----------------------------------|
| Instruction | Set the current phase to zero |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | FREQ > Reset phase delta display |
| Example | :PHASe:RESet :PHASe:REF |

3.5.2 [:SOURce]:POWer Subsystem

3.5.2.1 Level Display ([:SOURce]:POWer:POWer)

| | |
|-----------------|--|
| Command Format | [:SOURce]:POWer:POWer <power> [:SOURce]:POWer:POWer? |
| Instruction | Set the RF level display on parameter bar Get the RF level display from the parameter bar |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | Level Offset + Full power range |
| Return | Float, unit: dBm |
| Default | -130 dBm |
| Menu | Level |
| Example | POWer:POWer 2 POWer:POWer? |

3.5.2.2 Level ([:SOURce]:POWer)

| | |
|-----------------|--|
| Command Format | [:SOURce]:POWer <power> [:SOURce]:POWer? |
| Instruction | Set the RF output level Get the RF output level |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | Please refer to SSG5000A datasheet. |
| Return | Float, unit: dBm |
| Default | -130 dBm |
| Menu | LEVEL > Level |
| Example | POWer 2 :POWer? |

3.5.2.3 Level

([:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude])

| | |
|-----------------|--|
| Command | [:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] <power> |
| Format | [:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]? |
| Instruction | Set the RF output level Get the RF output level |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | Please refer to SSG5000A datasheet. |
| Return | Float, unit: dBm |
| Default | -130 dBm |
| Menu | LEVEL > Level |
| Example | POWer:LEVel -5 :POWer:LEVel? |

3.5.2.4 Level Offset ([:SOURce]:POWer:OFFSet)

| | |
|-----------------|--|
| Command | [:SOURce]:POWer:OFFSet <power> |
| Format | [:SOURce]:POWer:OFFSet? |
| Instruction | Set the RF offset level of the RF output Get the RF offset level of the RF output |
| Parameter Type | Float |
| Parameter Range | -100 dB ~ 100 dB |
| Return | Float, unit: dB |
| Default | 0 dB |
| Menu | LEVEL > Level Offset |
| Example | POWer:OFFSet 2 POWer:OFFSet? |

3.5.2.5 ALC State ([:SOURce]:POWer:ALC)

| | |
|-----------------|---|
| Command | [:SOURce]:POWer:ALC ON OFF AUTO |
| Format | [:SOURce]:POWer:ALC? |
| Instruction | Activate/deactivate automatic level control. Query ALC state |
| Parameter Type | Enumeration |
| Parameter Range | ON OFF AUTO ON Internal level control is permanently activated. OFF Internal level control is deactivated; Sample & Hold mode is activated. |

SIGLENT

| | |
|---------|---|
| | AUTO Internal level control is activated/deactivated automatically depending on the operating state. |
| Return | Enumeration |
| Default | AUTO |
| Menu | LEVEL > ALC State |
| Example | POWER:ALC ON POWER:ALC? |

3.5.2.6 Flatness List State ([:SOURCE]:CORRection[:FLATness])

| | |
|-----------------|---|
| Command | [[:SOURCE]:CORRection[:FLATness] ON OFF 1 0 |
| Format | [[:SOURCE]:CORRection[:FLATness]?] |
| Instruction | Activate/deactivate flatness correction list. |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | LEVEL > Flatness |
| Example | CORRection:FLATness ON :CORRection? |

3.5.2.7 Flatness List Add Row ([:SOURCE]:CORRection:FLATness:PAIR)

| | |
|-----------------|---|
| Command | [[:SOURCE]:CORRection:FLATness:PAIR <freq>, <power> |
| Format | |
| Instruction | Insert a new row in the flatness list. |
| Parameter Type | Float, Float |
| Parameter Range | Freq: Full freq range Power: Full power range |
| Return | None |
| Default | None |
| Menu | LEVEL > Flatness > Add |
| Example | CORRection:FLATness:PAIR 3 MHz, -3 |

3.5.2.8 Flatness List Delete Row ([:SOURCE]:CORRection:FLATness:DELeTe)

| | |
|----------------|---|
| Command | [[:SOURCE]:CORRection:FLATness:DELeTe <row> |
| Format | |
| Instruction | Delete the selected row in the flatness list. |
| Parameter Type | Integer |

| | |
|-----------------|--|
| Parameter Range | Less than the total count of the flatness. |
| Return | None |
| Default | None |
| Menu | LEVEL > Flatness > Delete |
| Example | CORRection:FLATness:DELeTe 0 |

3.5.2.9 Flatness List Count

([:SOURce]:CORRection:FLATness:COUNT?)

| | |
|-----------------|--|
| Command Format | [:SOURce]:CORRection:FLATness:COUNT? |
| Instruction | Indicates the total count of the number of elements in the flatness correction table |
| Parameter Type | None |
| Parameter Range | None |
| Return | Integer |
| Default | 0 |
| Menu | LEVEL > Flatness |
| Example | CORRection:FLATness:COUNT? |

3.5.2.10 Flatness List Store ([:SOURce]:CORRection:STORe)

| | |
|-----------------|--|
| Command Format | [:SOURce]:CORRection:STORe <file_name> |
| Instruction | Save the correction data in the list |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | LEVEL > Flatness > Save |
| Example | :CORRection:STORe "U-disk3/test.uflt" |

3.5.2.11 Flatness List Load ([:SOURce]:CORRection:LOAD)

| | |
|-----------------|---|
| Command Format | [:SOURce]:CORRection:LOAD <file_name> |
| Instruction | Load an existing flatness correction file |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |

| | |
|---------|--------------------------------------|
| Default | None |
| Menu | LEVEL > Flatness > Load |
| Example | :CORRection:LOAD “U-disk3/test.uflt” |

3.5.2.12 Flatness List Clear

([:SOURce]:CORRection:FLATness:PRESet)

| | |
|--------------------|--|
| Command Format | [:SOURce]:CORRection:FLATness:PRESet |
| Instruction | Clear the displayed flatness correction list |
| Parameter Type | None |
| Parameter Range | None |
| Default | None |
| Menu | LEVEL > Flatness > Clear |
| Example | :CORRection:FLATness:PRESet |

3.5.2.13 Flatness List Fill Type

([:SOURce]:CORRection:FLATness:FILL:TYPE)

| | |
|--------------------|---|
| Command Format | [:SOURce]:CORRection:FLATness:FILL:TYPE FLATness MANUal SWEEPlist [:SOURce]:CORRection:FLATness:FILL:TYPE? |
| Instruction | Set the Fill Type to generate flatness list. Get the Fill Type to generate flatness list. |
| Parameter Type | Enumeration |
| Parameter Range | FLATness MANUal SWEEPlist |
| Return | Enumeration |
| Default | FLATness |
| Menu | LEVEL > Flatness > Setting > Fill Type |
| Example | :CORRection:FLATness:FILL:TYPE FLATness :CORRection:FLATness:FILL:TYPE? |

3.5.2.14 Flatness List Start Freq

([:SOURce]:CORRection:FLATness:STARTfreq)

| | |
|-------------------|--|
| Command Format | [:SOURce]:CORRection:FLATness:STARTfreq <freq> [:SOURce]:CORRection:FLATness:STARTfreq? |
| Instruction | Set the start frequency when you want to fill the flatness list with the sensor and filling type is “Manual Step”. Get the start frequency when you want to fill the flatness list with the sensor and filling type is “Manual Step”. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter | Full frequency range |

| | |
|---------|---|
| Range | |
| Return | Float, unit: Hz |
| Default | Maximum frequency |
| Menu | LEVEL > Flatness > Setting > Fill Type > Manual Step > Start Freq |
| Example | :CORRection:FLATness:STARTfreq 200 MHz :CORRection:FLATness:STARTfreq? |

3.5.2.15 Flatness List Stop Freq ([:SOURce]:CORRection:FLATness:STOPfreq)

| | |
|-----------------|--|
| Command | :[:SOURce]:CORRection:FLATness:STOPfreq <freq> |
| Format | :[:SOURce]:CORRection:FLATness:STOPfreq? |
| Instruction | Set the stop frequency when you want to fill the flatness list with the sensor and filling type is “Manual Step”. Get the stop frequency when you want to fill the flatness list with the sensor and filling type is “Manual Step”. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | Full frequency range |
| Return | Float, unit: Hz |
| Default | Maximum frequency |
| Menu | LEVEL > Flatness > Setting > Fill Type > Manual Step > Stop Freq |
| Example | :CORRection:FLATness:STOPfreq 500 MHz :CORRection:FLATness:STOPfreq? |

3.5.2.16 Flatness List Fill Space ([:SOURce]:CORRection:FLATness:SPACE)

| | |
|-----------------|--|
| Command | :[:SOURce]:CORRection:FLATness:SPACE LINear LOGarithmic |
| Format | :[:SOURce]:CORRection:FLATness:SPACE? |
| Instruction | Set the fill space in Manual Step Fill Type. Get the fill space in Manual Step Fill Type. |
| Parameter Type | Enumeration |
| Parameter Range | LINear LOGarithmic |
| Return | Enumeration |
| Default | LINear |
| Menu | LEVEL > Flatness > Setting > Fill Type > Manual Step > Fill Space |
| Example | :CORRection:FLATness:SPACE LINear :CORRection:FLATness:SPACE? |

3.5.2.17 Flatness List Linear Step ([:SOURce]:CORRection:FLATness:LINStep)

| | |
|---------|---|
| Command | :[:SOURce]:CORRection:FLATness:LINStep <freq> |
|---------|---|

| | |
|-----------------|--|
| Format | [[:SOURce]:CORRection:FLATness:LINStep? |
| Instruction | Set the linear frequency step in Manual Step Fill Type. Get the linear frequency step in Manual Step Fill Type. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | None |
| Return | Float, unit: Hz |
| Default | None |
| Menu | LEVEL > Flatness > Setting > Fill Type > Manual Step > Step Linear |
| Example | :CORRection:FLATness:LINStep 200 MHz :CORRection:FLATness:LINStep? |

3.5.2.18 Flatness List Log Step ([[:SOURce]:CORRection:FLATness:LOGStep)

| | |
|-----------------|--|
| Command | [[:SOURce]:CORRection:FLATness:LOGStep <value> |
| Format | [[:SOURce]:CORRection:FLATness:LOGStep? |
| Instruction | Set the log frequency step in Manual Step Fill Type. Get the log frequency step in Manual Step Fill Type. |
| Parameter Type | Float, unit: % |
| Parameter Range | None |
| Return | Float, unit: % |
| Default | None |
| Menu | LEVEL > Flatness > Setting > Fill Type > Manual Step > Step Log |
| Example | :CORRection:FLATness:LOGStep 20 :CORRection:FLATness:LOGStep? |

3.5.2.19 Flatness List Points ([[:SOURce]:CORRection:FLATness:POINT)

| | |
|-----------------|--|
| Command | [[:SOURce]:CORRection:FLATness:POINT <points> |
| Format | [[:SOURce]:CORRection:FLATness:POINT? |
| Instruction | Set the points of flatness list in Manual Step Fill Type. Get the points of flatness list in Manual Step Fill Type. |
| Parameter Type | Integer |
| Parameter Range | 2 ~ 500 |
| Return | Integer |
| Default | 11 |
| Menu | LEVEL > Flatness > Setting > Fill Type > Manual Step > Points |
| Example | :CORRection:FLATness:POINT 5 :CORRection:FLATness:POINT? |

3.5.2.20 Fill Flatness with Sensor

([:SOURce]:CORRection:CSET:DATA[:SENSor][:POWer]:SONCe)

| | |
|-----------------|--|
| Command | [:SOURce]:CORRection:CSET:DATA[:SENSor][:POWer]:SONCe |
| Format | |
| Instruction | Fill the level values of the flatness list with the power meter. |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | LEVEL > Flatness > Setting > Fill Flatness with Sensor |
| Example | :CORRection:CSET:DATA:SONCe |

3.5.2.21 Level Control ([:SOURce]:POWer:SPC:STATe)

| | |
|-----------------|---|
| Command | [:SOURce]:POWer:SPC:STATe ON OFF 1 0 |
| Format | [:SOURce]:POWer:SPC:STATe? |
| Instruction | Activate/Deactivate power control using an external USB power sensor Get the level control state |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | SENSOR > Level Control |
| Example | POWer:SPC:STATe ON :POWer:SPC:STATe? |

3.5.2.22 Level Control (:SENSe[:POWer]:LEV:CTL:STATe)

| | |
|-----------------|---|
| Command | :SENSe[:POWer]:LEV:CTL:STATe ON OFF 1 0 |
| Format | :SENSe[:POWer]:LEV:CTL:STATe? |
| Instruction | Activate/Deactivate power control using an external USB power sensor Get the level control state |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | SENSOR > Level Control |
| Example | :SENSe:LEV:CTL:STATe OFF :SENSe:LEV:CTL:STATe? |

3.5.2.23 Target Level ([:SOURce]:POWer:SPC:TARGet)

| | |
|-----------------|--|
| Command | [:SOURce]:POWer:SPC:TARGet <power> |
| Format | [:SOURce]:POWer:SPC:TARGet? |
| Instruction | Set the nominal level expected at the input of the sensor Get the nominal level expected at the input of the sensor |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | -120 dBm ~ 20 dBm |
| Return | Float, unit: dBm |
| Default | 0 dBm |
| Menu | SENSOR > Level Control > Target Level |
| Example | POWer:SPC:TARGet 0 POWer:SPC:TARGet? |

3.5.2.24 Target Level (:SENSe[:POWer]:SPC:TARGet)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:SPC:TARGet <power> |
| Format | :SENSe[:POWer]:SPC:TARGet? |
| Instruction | Set the nominal level expected at the input of the sensor Get the nominal level expected at the input of the sensor |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | -120 dBm ~ 20 dBm |
| Return | Float, unit: dBm |
| Default | 0 dBm |
| Menu | SENSOR > Level Control > Target Level |
| Example | SENSe:SPC:TARGet -6 SENSe:SPC:TARGet? |

3.5.2.25 Level Limit ([:SOURce]:POWer:LIMit)

| | |
|-----------------|--|
| Command | [:SOURce]:POWer:LIMit <power> |
| Format | [:SOURce]:POWer:LIMit? |
| Instruction | Set the upper limit for the RF output power Get the upper limit for the RF output power |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | -120 dBm ~ 20 dBm |
| Return | Float, unit: dBm |
| Default | 0 dBm |
| Menu | SENSOR > Level Control > Level Limit |
| Example | POWer:LIMit 1 POWer:LIMit? |

3.5.2.26 Level Limit (:SENSe[:POWer]:LIMit)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:LIMit <power> |
| Format | :SENSe[:POWer]:LIMit? |
| Instruction | Set the upper limit for the RF output power Get the upper limit for the RF output power |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | -120 dBm ~ 20 dBm |
| Return | Float, unit: dBm |
| Default | 0 dBm |
| Menu | SENSOR > Level Control > Level Limit |
| Example | SENSe:LIMit 2 SENSe:LIMit? |

3.5.2.27 Catch Range ([:SOURce]:POWer:SPC:CRANge)

| | |
|-----------------|--|
| Command | [:SOURce]:POWer:SPC:CRANge <power> |
| Format | [:SOURce]:POWer:SPC:CRANge? |
| Instruction | Set the capture range of the control system Get the capture range of the control system |
| Parameter Type | Float, unit: dB |
| Parameter Range | 0 dB ~ 50 dB |
| Return | Float, unit: dB |
| Default | 0 dB |
| Menu | SENSOR > Level Control > Catch Range |
| Example | :POWer:SPC:CRANge 5 :POWer:SPC:CRANge? |

3.5.2.28 Catch Range (:SENSe[:POWer]:SPC:CRANge)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:SPC:CRANge <power> |
| Format | :SENSe[:POWer]:SPC:CRANge? |
| Instruction | Set the capture range of the control system Get the capture range of the control system |
| Parameter Type | Float, unit: dB |
| Parameter Range | 0 dB ~ 50 dB |
| Return | Float, unit: dB |
| Default | 0 dB |
| Menu | SENSOR > Level Control > Catch Range |
| Example | :SENSe:SPC:CRANge 10 :SENSe:SPC:CRANge? |

3.5.3 [:SOURce]:SWEep Subsystem

3.5.3.1 Sweep State ([:SOURce]:SWEep:STATe)

| | |
|-----------------|--|
| Command | [:SOURce]:SWEep:STATe OFF FREQuency LEVEl LEV_FREQ |
| Format | [:SOURce]:SWEep:STATe? |
| Instruction | Activate frequency or/and level sweep |
| Parameter Type | Enumeration |
| Parameter Range | OFF FREQuency LEVEl LEV_FREQ |
| Return | Enumeration |
| Default | OFF |
| Menu | SWEEP > Sweep State |
| Example | :SWEep:STATe OFF :SWEep:STATe? |

3.5.3.2 Sweep Type ([:SOURce]:SWEep:TYPE)

| | |
|-----------------|----------------------------------|
| Command | [:SOURce]:SWEep:TYPE LIST STEP |
| Format | [:SOURce]:SWEep:TYPE? |
| Instruction | Set sweep type Get sweep type |
| Parameter Type | Enumeration |
| Parameter Range | LIST STEP |
| Return | Enumeration |
| Default | STEP |
| Menu | SWEEP > Step Sweep / List Sweep |
| Example | :SWEep:TYPE STEP :SWEep:TYPE? |

3.5.3.3 Start Frequency ([:SOURce]:SWEep:STEP:START:FREQuency)

| | |
|-----------------|--|
| Command | [:SOURce]:SWEep:STEP:START:FREQuency <freq> |
| Format | [:SOURce]:SWEep:STEP:START:FREQuency? |
| Instruction | Set the start frequency for the sweep mode Get the start frequency for the sweep mode |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | Full frequency range. |
| Return | Float, unit: Hz |
| Default | Maximum frequency |
| Menu | SWEEP > Step Sweep > Start Freq |
| Example | :SWEep:STEP:START:FREQuency 1 GHz |

| | |
|--|------------------------------|
| | :SWEep:STEP:START:FREQuency? |
|--|------------------------------|

3.5.3.4 Stop Frequency ([:SOURce]:SWEep:STEP:STOP:FREQuency)

| | |
|-----------------|--|
| Command | [[:SOURce]:SWEep:STEP:STOP:FREQuency <freq> |
| Format | [[:SOURce]:SWEep:STEP:STOP:FREQuency? |
| Instruction | Set the stop frequency for the sweep mode Get the stop frequency for the sweep mode |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | Full frequency range. |
| Return | Float, unit: Hz |
| Default | Maximum frequency |
| Menu | SWEEP > Step Sweep > Stop Freq |
| Example | :SWEep:STEP:STOP:FREQuency 1 GHz :SWEep:STEP:STOP:FREQuency? |

3.5.3.5 Start Level ([:SOURce]:SWEep:STEP:START:LEVEl)

| | |
|-----------------|--|
| Command | [[:SOURce]:SWEep:STEP:START:LEVEl <level> |
| Format | [[:SOURce]:SWEep:STEP:START:LEVEl? |
| Instruction | Set the start level for the sweep mode Get the start level for the sweep mode |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default: dBm |
| Parameter Range | Full level range. |
| Return | Float, unit: dBm |
| Default | -130 dBm |
| Menu | SWEEP > Step Sweep > Start Level |
| Example | :SWEep:STEP:START:LEVEl 0 dBm :SWEep:STEP:START:LEVEl? |

3.5.3.6 Stop Level ([:SOURce]:SWEep:STEP:STOP:LEVEl)

| | |
|-----------------|--|
| Command | [[:SOURce]:SWEep:STEP:STOP:LEVEl <level> |
| Format | [[:SOURce]:SWEep:STEP:STOP:LEVEl? |
| Instruction | Set the stop level for the sweep mode Get the stop level for the sweep mode |
| Parameter Type | Float, unit: dBm, dBuV, uV, mV, V, nW, uW, mW, W, Default dBm |
| Parameter Range | Full level range. |
| Return | Float, unit: dBm |
| Default | -130 dBm |
| Menu | SWEEP > Step Sweep > Stop Level |

| | |
|---------|------------------------------|
| Example | :SWEep:STEP:STOP:LEVel 0 dBm |
| | :SWEep:STEP:STOP:LEVel? |

3.5.3.7 Dwell Time ([:SOURce]:SWEep:STEP:DWEL1)

| | |
|-----------------|--|
| Command | [[:SOURce]:SWEep:STEP:DWEL1 <time> |
| Format | [[:SOURce]:SWEep:STEP:DWEL1? |
| Instruction | Set the duration of the individual sweep steps Get the duration of the individual sweep steps |
| Parameter Type | Float, unit: ns, us, ms, s |
| Parameter Range | 10 ms ~ 100 s |
| Return | Float, unit: s |
| Default | 30 ms |
| Menu | SWEEP > Step Sweep > Dwell Time |
| Example | :SWEep:STEP:DWEL1 20 ms :SWEep:STEP:DWEL1? |

3.5.3.8 Sweep Points ([:SOURce]:SWEep:STEP:POINTs)

| | |
|-----------------|--|
| Command | [[:SOURce]:SWEep:STEP:POINTs <points> |
| Format | [[:SOURce]:SWEep:STEP:POINTs? |
| Instruction | Set the number of steps in an RF sweep Get the number of steps in an RF sweep |
| Parameter Type | Integer |
| Parameter Range | 2 ~ 65535 |
| Return | Integer |
| Default | 11 |
| Menu | SWEEP > Step Sweep > Sweep Points |
| Example | :SWEep:STEP:POINTs 2 :SWEep:STEP:POINTs? |

3.5.3.9 Sweep Shape ([:SOURce]:SWEep:STEP:SHAPE)

| | |
|-----------------|---|
| Command | [[:SOURce]:SWEep:STEP:SHAPE TRIangle SAWtooth |
| Format | [[:SOURce]:SWEep:STEP:SHAPE? |
| Instruction | Select the waveform shape of the sweep signal Get the waveform shape of the sweep signal |
| Parameter Type | Enumeration |
| Parameter Range | TRIangle SAWtooth |
| Return | Enumeration |
| Default | SAWTooth |

| | |
|---------|--|
| Menu | SWEEP > Step Sweep > Sweep Shape |
| Example | :SWEep:STEP:SHAPE TRIangle :SWEep:STEP:SHAPE? |

3.5.3.10 Sweep Space ([:SOURce]:SWEep:STEP:SPACe)

| | |
|-----------------|---|
| Command | [:SOURce]:SWEep:STEP:SPACe LINear LOGarithmic |
| Format | |
| Instruction | Select the sweep spacing Get the sweep spacing |
| Parameter Type | Enumeration |
| Parameter Range | LINear LOGarithmic |
| Return | Enumeration |
| Default | LINear |
| Menu | SWEEP > Step Sweep > Sweep Space |
| Example | :SWEep:STEP:SPACe LOGarithmic :SWEep:STEP:SPACe? |

3.5.3.11 Sweep Step in Linear Sweep Space ([:SOURce]:SWEep[:FREQuency]:STEP[:LINear])

| | |
|-----------------|--|
| Command | [:SOURce]:SWEep[:FREQuency]:STEP[:LINear] <freq> |
| Format | |
| Instruction | Set the sweep step in linear sweep space. Get the sweep step in linear sweep space. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | None |
| Return | Float, unit: Hz |
| Default | 0 |
| Menu | SWEEP > Step Sweep > Freq Step Linear |
| Example | :SWEep:STEP 200 MHz :SWEep:STEP? |

3.5.3.12 Sweep Step in Log Sweep Space ([:SOURce]:SWEep[:FREQuency]:STEP:LOGarithmic)

| | |
|----------------|--|
| Command | [:SOURce]:SWEep[:FREQuency]:STEP:LOGarithmic <value> |
| Format | |
| Instruction | Set the sweep step in logarithmic sweep space. Get the sweep step in logarithmic sweep space. |
| Parameter Type | Float, unit: % |
| Parameter | None |

| | |
|---------|--|
| Range | |
| Return | Float, unit: % |
| Default | 0 |
| Menu | SWEEP > Step Sweep > Freq Step Log |
| Example | :SWEep:STEP:LOGarithmic 20 :SWEep:STEP:LOGarithmic? |

3.5.3.13 Sweep List Add Row ([:SOURce]:SWEep:LIST:ADDList)

| | |
|-----------------|--|
| Command Format | [:SOURce]:SWEep:LIST:ADDList <freq>,<level>,<time> |
| Instruction | Insert a new row to the list |
| Parameter Type | Freq: Float, unit: Hz, kHz, MHz, GHz, Default “Hz” Level: Float, unit: dBm Time: Float, unit: ns, us, ms, s, Default “s” |
| Parameter Range | Full frequency range, full frequency range, 10.0 ms ~ 100.0 s |
| Return | None |
| Default | None |
| Menu | SWEEP > List Sweep > Add |
| Example | :SWEep:LIST:ADDList 1 GHz,0 dBm,1 s |

3.5.3.14 Sweep List Delete Row ([:SOURce]:SWEep:LIST:DELeTe)

| | |
|-----------------|--|
| Command Format | [:SOURce]:SWEep:LIST:DELeTe <row> |
| Instruction | Delete the sweep list pair |
| Parameter Type | Integer |
| Parameter Range | 1 to the full count of the sweep list. |
| Return | None |
| Default | None |
| Menu | SWEEP > List Sweep > Delete |
| Example | :SWEep:LIST:DELeTe 1 |

3.5.3.15 Sweep List Edit ([:SOURce]:SWEep:LIST:CHANGe)

| | |
|-----------------|---|
| Command Format | [:SOURce]:SWEep:LIST:CHANGe <row>,<freq>,<power>,<time> |
| Instruction | Edit sweep list pair value |
| Parameter Type | Row: Integer, Freq: Float, unit: Hz, kHz, MHz, GHz, Power: Float, unit: dBm, dBmV, dBuV, V, W, Default: dBm, Time: Float, unit: ns, us, ms, s, Default “s” |
| Parameter Range | Row: 1 ~ count of pair. Freq: Full frequency range. |

| | |
|---------|--|
| | Power: Full level range. time: 10 ms ~ 100 s. |
| Return | None |
| Default | None |
| Menu | SWEEP > List Sweep |
| Example | :SWEep:LIST:CHANGe 1,1 GHz,1 dBm, 1 s |

3.5.3.16 Sweep List Row Count ([:SOURce]:SWEep:LIST:CPOint?)

| | |
|-----------------|---------------------------------|
| Command Format | [:SOURce]:SWEep:LIST:CPOint? |
| Instruction | Get how many rows in sweep list |
| Parameter Type | None |
| Parameter Range | None |
| Return | Float |
| Default | 1 |
| Menu | SWEEP > List Sweep |
| Example | :SWEep:LIST:CPOint? |

3.5.3.17 Show Sweep List ([:SOURce]:SWEep:LIST:LIST?)

| | |
|-----------------|---|
| Command Format | [:SOURce]:SWEep:LIST:LIST? <begin_ row>,<end_row> |
| Instruction | View starting row to end row data |
| Parameter Type | Integer, Integer |
| Parameter Range | 1 to count of sweep list. |
| Return | String |
| Default | None |
| Menu | SWEEP > List Sweep |
| Example | :SWEep:LIST:LIST? 1,3 |

3.5.3.18 Sweep List Clear ([:SOURce]:SWEep:LIST:INITialize:PRESet)

| | |
|-----------------|---|
| Command Format | [:SOURce]:SWEep:LIST:INITialize:PRESet |
| Instruction | Restore the scan list of the factory default settings |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |

SIGLENT

| | |
|---------|------------------------------|
| Menu | SWEEP > List Sweep > Clear |
| Example | SWEEp:LIST:INITialize:PRESet |

3.5.3.19 Sweep List Initialize From Step ([:SOURce]:SWEEp:LIST:INITialize:FSTep)

| | |
|-----------------|---|
| Command Format | [:SOURce]:SWEEp:LIST:INITialize:FSTep |
| Instruction | Regenerate the sweep list based on the data points of the current step sweep settings |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | SWEEP > List Sweep > Import |
| Example | SWEEp:LIST:INITialize:FSTep |

3.5.3.20 Sweep List Load ([:SOURce]:SWEEp:LOAD)

| | |
|-----------------|---------------------------------------|
| Command Format | [:SOURce]:CORRection:LOAD <file_name> |
| Instruction | Load existing sweep list file |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | SWEEP > List Sweep > Load |
| Example | :SWEEp:LOAD "U-disk3/test.lsw" |

3.5.3.21 Sweep List Store ([:SOURce]:SWEEp:STORe)

| | |
|-----------------|--|
| Command Format | [:SOURce]:CORRection:STORe <file_name> |
| Instruction | Save the sweep data in the list |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | SWEEP > List Sweep > Save |
| Example | :SWEEp:STORe "U-disk3/test.lsw" |

3.5.3.22 Sweep Direction ([:SOURce]:SWEep:DIRect)

| | |
|-----------------|-------------------------------------|
| Command | [:SOURce]:SWEep:DIRect FWD REV |
| Format | [:SOURce]:SWEep:DIRect? |
| Instruction | Select the direction for sweep |
| Parameter Type | Enumeration |
| Parameter Range | FWD REV |
| Return | Enumeration |
| Default | FWD |
| Menu | SWEEP > Direction |
| Example | :SWEep:DIRect REV :SWEep:DIRect? |

3.5.3.23 Sweep Mode ([:SOURce]:SWEep:MODE)

| | |
|-----------------|--|
| Command | [:SOURce]:SWEep:MODE CONTInue SINGle |
| Format | [:SOURce]:SWEep:MODE? |
| Instruction | Set the cycle mode of the sweep Get the cycle mode of the sweep |
| Parameter Type | Enumeration |
| Parameter Range | CONTInue SINGle |
| Return | Enumeration |
| Default | CONTInue |
| Menu | SWEEP > Sweep Mode |
| Example | :SWEep:MODE SINGle :SWEep:MODE? |

3.5.3.24 Execute Single Sweep ([:SOURce]:SWEep:EXECute)

| | |
|-----------------|---|
| Command | [:SOURce]:SWEep:EXECute |
| Format | |
| Instruction | Execute one single sweep when the sweep mode is Single. |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | SWEEP > Execute single sweep |
| Example | :SWEep:EXECute |

3.5.3.25 Trigger Mode ([:SOURce]:SWEep:SWEep:TRIGger:TYPE)

| | |
|-----------------|---|
| Command | [:SOURce]:SWEep:SWEep:TRIGger:TYPE AUTO KEY BUS EXT |
| Format | [:SOURce]:SWEep:SWEep:TRIGger:TYPE? |
| Instruction | Select the trigger mode Get the trigger mode |
| Parameter Type | Enumeration |
| Parameter Range | AUTO KEY BUS EXT |
| Return | Enumeration |
| Default | AUTO |
| Menu | SWEEP > Trigger Mode |
| Example | :SWEep:SWEep:TRIGger:TYPE KEY :SWEep:SWEep:TRIGger:TYPE? |

3.5.3.26 Point Trigger ([:SOURce]:SWEep:POINt:TRIGger:TYPE)

| | |
|-----------------|---|
| Command | [:SOURce]:SWEep:POINt:TRIGger:TYPE AUTO KEY BUS EXT |
| Format | [:SOURce]:SWEep:POINt:TRIGger:TYPE? |
| Instruction | Select the point trigger Get the point trigger |
| Parameter Type | Enumeration |
| Parameter Range | AUTO KEY BUS EXT |
| Return | Enumeration |
| Default | AUTO |
| Menu | SWEEP > Point Trigger |
| Example | :SWEep:POINt:TRIGger:TYPE KEY :SWEep:POINt:TRIGger:TYPE? |

3.5.3.27 Bus Trigger ([:SOURce]:*TRG)

| | |
|-----------------|---|
| Command | [:SOURce]:*TRG |
| Format | |
| Instruction | When the trigger mode or point trigger mode is Bus, send this command to make the signal source start sweeping. |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | None |
| Example | *TRG |

3.5.3.28 Trigger Slope ([:SOURce]:INPut:TRIGger:SLOPe)

| | |
|-----------------|---|
| Command Format | [:SOURce]:INPut:TRIGger:SLOPe POSitive NEGative [:SOURce]:INPut:TRIGger:SLOPe? |
| Instruction | Select the trigger slope when the trigger mode or point trigger is EXT. Get the trigger slope when the trigger mode or point trigger is EXT. |
| Parameter Type | Enumeration |
| Parameter Range | POSitive NEGative |
| Return | Enumeration |
| Default | POSitive |
| Menu | SWEEP > Trigger Slope |
| Example | :INPut:TRIGger:SLOPe NEGative :INPut:TRIGger:SLOPe? |

3.5.3.29 Get Sweep Point ([:SOURce]:SWEep:CURRent:Data?)

| | |
|-----------------|--|
| Command Format | [:SOURce]:SWEep:CURRent:Data? |
| Instruction | Get the currently sweep point. The format is: index, {freq, level, time} |
| Parameter Type | None |
| Parameter Range | None |
| Return | String Index: interger freq: Hz level: dBm time: s |
| Default | None |
| Menu | None |
| Example | :SWEep:CURRent:Data? Return: 1, {1e+09, -5, 0.03} |

3.5.4 [:SOURce]:MODulation Subsystem

3.5.4.1 Modulation State ([:SOURce]:MODulation)

| | |
|----------------|--|
| Command Format | [:SOURce]:MODulation ON OFF 1 0 [:SOURce]:MODulation? |
| Instruction | Switch modulation on and off Get the modulation state |
| Parameter | Boolean |

SIGLENT

| | |
|-----------------|-------------------------------|
| Type | |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | MOD |
| Example | MODulation ON :MODulation? |

3.5.4.2 Modulation State (:OUTPut:MODulation[:STATe])

| | |
|-----------------|--|
| Command | :OUTPut:MODulation[:STATe] ON OFF 1 0 |
| Format | :OUTPut:MODulation[:STATe]? |
| Instruction | Switch modulation on and off Get the modulation state |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | MOD |
| Example | :OUTPut:MODulation ON :OUTPut:MODulation? |

3.5.5 [:SOURce]:AM Subsystem

3.5.5.1 AM State ([:SOURce]:AM:STATe)

| | |
|-----------------|---|
| Command | [:SOURce]:AM:STATe ON OFF 1 0 |
| Format | [:SOURce]:AM:STATe? |
| Instruction | Activate/Deactivate amplitude modulation (AM) Get the AM state |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | AM > AM State |
| Example | :AM:STATe ON :AM:STATe? |

3.5.5.2 AM Shape ([:SOURce]:AM:WAVEform)

| | |
|---------|-----------------------------------|
| Command | [:SOURce]:AM:WAVEform SINE SQUAre |
| Format | [:SOURce]:AM:WAVEform? |

| | |
|-----------------|--|
| Instruction | Set the AM modulation waveform Get the AM modulation waveform |
| Parameter Type | Enumeration |
| Parameter Range | SINE SQUAre |
| Return | Enumeration |
| Default | SINE |
| Menu | AM > AM Shape |
| Example | :AM:WAVEform SINE :AM:WAVEform? |

3.5.5.3 AM Source ([:SOURce]:AM:SOURce)

| | |
|-----------------|---|
| Command Format | [[:SOURce]:AM:SOURce INTernal EXTernal INT,EXT [:SOURce]:AM:SOURce? |
| Instruction | Select the modulation signal source for amplitude modulation Get the AM source |
| Parameter Type | Enumeration |
| Parameter Range | INTernal EXTernal INT,EXT |
| Return | Enumeration |
| Default | INTernal |
| Menu | AM > AM Source |
| Example | :AM:SOURce EXTernal :AM:SOURce? |

3.5.5.4 AM Depth ([:SOURce]:AM:DEPTh)

| | |
|-----------------|---|
| Command Format | [[:SOURce]:AM:DEPTh <value> [:SOURce]:AM:DEPTh? |
| Instruction | Set the overall modulation depth of the amplitude modulation in percent Get the AM depth |
| Parameter Type | Float |
| Parameter Range | 0.1 % ~ 100 % |
| Return | Float |
| Default | 50 % |
| Menu | AM > AM Depth |
| Example | :AM:DEPTh 0.2 :AM:DEPTh? |

3.5.5.5 AM Rate ([:SOURce]:AM:FREQuency)

| | |
|-----------------|--|
| Command | [:SOURce]:AM:FREQuency <value> |
| Format | [:SOURce]:AM:FREQuency? |
| Instruction | Set the AM modulation frequency Get the AM modulation frequency |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | Sine: 0.01 Hz ~ 100 kHz Square: 0.01 Hz ~ 20 kHz |
| Return | Float, unit: Hz |
| Default | 1 kHz |
| Menu | AM > AM Rate |
| Example | :AM:FREQuency 10 kHz :AM:FREQuency? |

3.5.5.6 AM Sensitivity ([:SOURce]:AM:SENSitivity?)

| | |
|-----------------|--|
| Command | [:SOURce]:AM:SENSitivity? |
| Format | |
| Instruction | Query the input sensitivity of the external modulation input in /V |
| Parameter Type | None |
| Parameter Range | None |
| Return | Float, unit: /V |
| Default | 0 %/V |
| Menu | AM > AM Sensitivity |
| Example | AM:SENSitivity? |

3.5.6 [:SOURce]:FM Subsystem

3.5.6.1 FM State ([:SOURce]:FM:STATe)

| | |
|-----------------|---|
| Command | [:SOURce]:FM:STATe ON OFF 1 0 |
| Format | [:SOURce]:FM:STATe? |
| Instruction | Activate/Deactivate frequency modulation (FM) Get the FM state |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | FM > FM State |
| Example | :FM:STATe ON :FM:STATe? |

3.5.6.2 FM Source ([:SOURce]:FM:SOURce)

| | |
|-----------------|---|
| Command Format | [:SOURce]:FM:SOURce INT1 INT2 INT1,INT2 EXTErnal INT1,EXT DUAL [:SOURce]:FM:SOURce? |
| Instruction | Select the modulation signal source for frequency modulation (FM) Get the FM Source |
| Parameter Type | Enumeration |
| Parameter Range | INT1 INT2 INT1,INT2 EXTErnal INT1,EXT DUAL |
| Return | Enumeration |
| Default | INTernal |
| Menu | FM > FM Source |
| Example | :FM:SOURce EXTErnal :FM:SOURce? |

3.5.6.3 FM Shape1 ([:SOURce]:FM1:WAVEform)

| | |
|-----------------|--|
| Command Format | [:SOURce]:FM1:WAVEform SINE SQUAre SAWTooth TRIangle [:SOURce]:FM1:WAVEform? |
| Instruction | Selects the shape of the FM waveform1 when the FM Source includes Int1. Get the shape of the FM waveform1 when the FM Source includes Int1. |
| Parameter Type | Enumeration |
| Parameter Range | SINE SQUAre SAWTooth TRIangle |
| Return | Enumeration |
| Default | SINE |
| Menu | FM > FM Shape1 |
| Example | :FM1:WAVEform SQUAre :FM1:WAVEform? |

3.5.6.4 FM Deviation1 ([:SOURce]:FM1:DEViation)

| | |
|-----------------|--|
| Command Format | [:SOURce]:FM1:DEViation <value> [:SOURce]:FM1:DEViation? |
| Instruction | Set the deviation value of the FM waveform1 when the FM Source includes Int1. Get the deviation value of the FM waveform1 when the FM Source includes Int1. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz |
| Parameter Range | 1 Hz ~ N*1 MHz The Value of N please refer to SSG5000A datasheet. |
| Return | Float, unit: Hz |
| Default | 100 kHz |
| Menu | FM > FM Deviation1 |

| | |
|---------|---|
| Example | :FM1:DEVIation 500 kHz :FM1:DEVIation? |
|---------|---|

3.5.6.5 FM Rate1 ([:SOURce]:FM1:FREQuency)

| | |
|-----------------|--|
| Command | [:SOURce]:FM1:FREQuency <value> |
| Format | [:SOURce]:FM1:FREQuency? |
| Instruction | Set the modulation frequency of the FM waveform1 when the FM Source includes Int1. Get the modulation frequency of the FM waveform1 when the FM Source includes Int1. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | SQUAre SAWTooth TRIangle: 0.01 Hz ~ 20 kHz SINE: 0.01 Hz ~ 100 kHz |
| Return | Float, unit: Hz |
| Default | 10 kHz |
| Menu | FM > FM Rate1 |
| Example | :FM1:FREQuency 40 kHz :FM1:FREQuency? |

3.5.6.6 FM Phase1 ([:SOURce]:FM1:PHASe)

| | |
|-----------------|--|
| Command | [:SOURce]:FM1:PHASe <value> |
| Format | [:SOURce]:FM1:PHASe? |
| Instruction | Set the phase of the FM waveform1 when the FM Source is Int1+Int2 or Dual. Get the phase of the FM waveform1 when the FM Source is Int1+Int2 or Dual. |
| Parameter Type | Float, unit: deg or rad, Default “deg” |
| Parameter Range | -360° ~ +360° |
| Return | Float, unit: deg |
| Default | 0 |
| Menu | FM > FM Phase1 |
| Example | :FM1:PHASe -30 deg :FM1:PHASe? |

3.5.6.7 FM Shape2 ([:SOURce]:FM2:WAVEform)

| | |
|----------------|--|
| Command | [:SOURce]:FM2:WAVEform SINE SQUAre SAWTooth TRIangle |
| Format | [:SOURce]:FM2:WAVEform? |
| Instruction | Selects the shape of the FM waveform2 when the FM Source includes Int2. Get the shape of the FM waveform2 when the FM Source includes Int2. |
| Parameter Type | Enumeration |

| | |
|-----------------|--|
| Parameter Range | SINE SQUAre SAWTooth TRIangle |
| Return | Enumeration |
| Default | SINE |
| Menu | FM > FM Shape2 |
| Example | :FM2:WAVEform SAWTooth :FM2:WAVEform? |

3.5.6.8 FM Deviation2 ([:SOURce]:FM2:DEViation)

| | |
|-----------------|--|
| Command | [:SOURce]:FM2:DEViation <value> [:SOURce]:FM2:DEViation? |
| Format | |
| Instruction | Set the deviation value of the FM waveform2 when the FM Source includes Int2. Get the deviation value of the FM waveform2 when the FM Source includes Int2. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz |
| Parameter Range | 1 Hz ~ N*1 MHz The Value of N please refer to SSG5000A datasheet. |
| Return | Float, unit: Hz |
| Default | 100 kHz |
| Menu | FM > FM Deviation2 |
| Example | :FM2:DEViation 600 kHz :FM2:DEViation? |

3.5.6.9 FM Rate2 ([:SOURce]:FM2:FREQuency)

| | |
|-----------------|--|
| Command | [:SOURce]:FM2:FREQuency <value> [:SOURce]:FM2:FREQuency? |
| Format | |
| Instruction | Set the modulation frequency of the FM waveform2 when the FM Source includes Int2. Get the modulation frequency of the FM waveform2 when the FM Source includes Int2. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | SQUAre SAWTooth TRIangle: 0.01 Hz ~ 20 kHz SINE: 0.01 Hz ~ 100 kHz |
| Return | Float, unit: Hz |
| Default | 10 kHz |
| Menu | FM > FM Rate2 |
| Example | :FM2:FREQuency 15 kHz :FM2:FREQuency? |

3.5.6.10 FM Phase2 ([:SOURce]:FM2:PHASe)

| | |
|---------|---|
| Command | [:SOURce]:FM2:PHASe <value> [:SOURce]:FM2:PHASe? |
| Format | |

| | |
|-----------------|--|
| Instruction | Set the phase of the FM waveform2 when the FM Source is Int1+Int2 or Dual. Get the phase of the FM waveform2 when the FM Source is Int1+Int2 or Dual. |
| Parameter Type | Float, unit: deg or rad, Default “deg” |
| Parameter Range | -360° ~ +360° |
| Return | Float, unit: deg |
| Default | 0 |
| Menu | FM > FM Phase2 |
| Example | :FM2:PHASe 45 deg :FM2:PHASe? |

3.5.6.11 Int1 Proportion ([:SOURce]:FM1:PROPortion)

| | |
|-----------------|--|
| Command Format | [[:SOURce]:FM1:PROPortion <value> [:SOURce]:FM1:PROPortion? |
| Instruction | Set the proportion of the FM waveform1 when the FM Source is Dual. Get the proportion of the FM waveform1 when the FM Source is Dual. |
| Parameter Type | Float |
| Parameter Range | 0 ~ 100% |
| Return | Float |
| Default | 50% |
| Menu | FM > Int1 Proportion |
| Example | :FM1:PROPortion 0.6 :FM1:PROPortion? |

3.5.6.12 FM Sensitivity1 ([:SOURce]:FM1:SENSitivity?)

| | |
|-----------------|---|
| Command Format | [[:SOURce]:FM1:SENSitivity? |
| Instruction | Displays the input sensitivity of the FM EXT input in Hz/V when the FM Source includes Ext. |
| Parameter Type | None |
| Parameter Range | None |
| Return | Float unit: Hz/V |
| Default | 0 Hz/V |
| Menu | FM > FM Sensitivity1 |
| Example | FM1:SENSitivity? |

3.5.7 [:SOURce]:PM Subsystem

3.5.7.1 PM State ([:SOURce]:PM:STATe)

| | |
|-----------------|---|
| Command | [:SOURce]:PM:STATe ON OFF 1 0 |
| Format | [:SOURce]:PM:STATe? |
| Instruction | Activate/Deactivate phase modulation (PM) Get the PM state |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | PM > PM State |
| Example | :PM:STATe ON :PM:STATe? |

3.5.7.2 PM Shape ([:SOURce]:PM:WAVEform)

| | |
|-----------------|--|
| Command | [:SOURce]:PM:WAVEform SINE SQUAre |
| Format | [:SOURce]:PM:WAVEform? |
| Instruction | Selects the shape of PM Get the shape of PM |
| Parameter Type | Enumeration |
| Parameter Range | SINE SQUAre |
| Return | Enumeration |
| Default | SINE |
| Menu | PM > PM Shape |
| Example | :PM:WAVEform SINE :PM:WAVEform? |

3.5.7.3 PM Source ([:SOURce]:PM:SOURce)

| | |
|-----------------|--|
| Command | [:SOURce]:PM:SOURce INTeRnal EXTeRnal INT,EXT |
| Format | [:SOURce]:PM:SOURce? |
| Instruction | Select the modulation signal source for phase modulation (PM) Get the PM source |
| Parameter Type | Enumeration |
| Parameter Range | INTeRnal EXTeRnal INT,EXT |
| Return | Enumeration |
| Default | INTeRnal |
| Menu | PM > PM Source |

| | |
|---------|------------------------------------|
| Example | :PM:SOURce EXternal :PM:SOURce? |
|---------|------------------------------------|

3.5.7.4 PM Deviation ([:SOURce]:PM:DEVIation)

| | |
|-----------------|--|
| Command | [:SOURce]:PM:DEVIation <value> |
| Format | [:SOURce]:PM:DEVIation? |
| Instruction | Set the modulation deviation of the phase modulation (PM) Get the modulation deviation of the phase modulation (PM) |
| Parameter Type | Float, unit: rad |
| Parameter Range | 0.01 rad ~ N*5 rad The value of N please refer to SSG5000A datasheet. |
| Return | Float, unit: rad |
| Default | 1 rad |
| Menu | PM > PM Deviation |
| Example | :PM:DEVIation 2 :PM:DEVIation? |

3.5.7.5 PM Rate ([:SOURce]:PM:FREQuency)

| | |
|-----------------|---|
| Command | [:SOURce]:PM:FREQuency <value> |
| Format | [:SOURce]:PM:FREQuency? |
| Instruction | Set the phase modulation (PM) frequency Get the phase modulation (PM) frequency |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | INTernal: SQUAre: 0.01 Hz ~ 20 kHz, SINE: 0.01 Hz ~ 100 kHz INT+EXT: SQUAre: 0.01 Hz ~ 20 kHz, SINE: 0.01 Hz ~ 100 kHz |
| Return | Float, unit: Hz |
| Default | 10 kHz |
| Menu | PM > PM Rate |
| Example | :PM:FREQuency 10 kHz :PM:FREQuency? |

3.5.7.6 PM Sensitivity ([:SOURce]:PM:SENSitivity?)

| | |
|-----------------|---|
| Command | [:SOURce]:PM:SENSitivity? |
| Format | |
| Instruction | Query the input sensitivity of the EXT MOD input in rad/v |
| Parameter Type | None |
| Parameter Range | None |
| Return | Float, unit: rad/V |
| Default | 0 rad/V |
| Menu | PM > PM Sensitivity |

| | |
|---------|-----------------|
| Example | PM:SENSitivity? |
|---------|-----------------|

3.5.8 [:SOURce]:PULM Subsystem

3.5.8.1 Pulse State ([:SOURce]:PULM:STATe)

| | |
|-----------------|---|
| Command | [:SOURce]:PULM:STATe ON OFF 1 0 |
| Format | [:SOURce]:PULM:STATe? |
| Instruction | Activate/Deactivate the pulse modulation Get the state of pulse modulation |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | PULSE > Pulse State |
| Example | PULM:STAT ON :PULM:STATe? |

3.5.8.2 Pulse Out ([:SOURce]:PULM:OUT:STATe)

| | |
|-----------------|---|
| Command | [:SOURce]:PULM:OUT:STATe ON OFF 1 0 |
| Format | [:SOURce]:PULM:OUT:STATe? |
| Instruction | Configures the signal at the PULSE OUT connector Get the Pulse Output status |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | PULSE > Pulse Out |
| Example | PULM:OUT:STATe ON :PULM:OUT:STATe? |

3.5.8.3 Pulse Source ([:SOURce]:PULM:SOURce)

| | |
|-----------------|---|
| Command | [:SOURce]:PULM:SOURce INTERNAL EXTERNAL |
| Format | [:SOURce]:PULM:SOURce? |
| Instruction | Select the source for the pulse modulation signal Get the source for the pulse modulation signal |
| Parameter Type | Enumeration |
| Parameter Range | INTERNAL EXTERNAL |
| Return | Enumeration |

SIGLENT

| | |
|---------|-------------------------------------|
| Default | INTernal |
| Menu | PULSE > Pulse Source |
| Example | PULM:SOUR INTernal :PULM:SOURce? |

3.5.8.4 Pulse Source ([:SOURce]:PULM:SOURce:INT)

| | |
|-----------------|---|
| Command | [:SOURce]:PULM:SOURce:INT INTernal EXTernal |
| Format | [:SOURce]:PULM:SOURce:INT? |
| Instruction | Select the source for the pulse modulation signal Get the source for the pulse modulation signal |
| Parameter Type | Enumeration |
| Parameter Range | INTernal EXTernal |
| Return | Enumeration |
| Default | INTernal |
| Menu | PULSE > Pulse Source |
| Example | :PULM:SOURce:INT EXTernal :PULM:SOURce:INT? |

3.5.8.5 Pulse Out Polarity ([:SOURce]:PULM:POLarity)

| | |
|-----------------|---|
| Command | [:SOURce]:PULM:POLarity NORMal INVerted |
| Format | [:SOURce]:PULM:POLarity? |
| Instruction | Set the period of the generated pulse. The period determines the repetition frequency of the internal signal Get the period of the generated pulse |
| Parameter Type | Enumeration |
| Parameter Range | NORMal INVerted |
| Return | Enumeration |
| Default | NORMal |
| Menu | PULSE > Pulse Out Polarity |
| Example | PULM:POL INV :PULM:POLarity? |

3.5.8.6 Pulse Mode ([:SOURce]:PULM:MODE)

| | |
|----------------|--|
| Command | [:SOURce]:PULM:MODE SINGle DOUBle PTRain |
| Format | [:SOURce]:PULM:MODE? |
| Instruction | Set the mode of the pulse generator Get the mode of the pulse generator |
| Parameter Type | Enumeration |
| Parameter | SINGle DOUBle PTRain |

| | |
|---------|---|
| Range | <p>SINGle Enables single pulse generation.</p> <p>DOUBle Enables double pulse generation. The two pulses are generated in one pulse period.</p> <p>PTRain A user-defined pulse train is generated. The pulse train is defined by value pairs of on and off times that can be entered in a pulse train list.</p> |
| Return | Enumeration |
| Default | SINGle |
| Menu | PULSE > Pulse Mode |
| Example | PULM:MODE DOUB PULM:MODE? |

3.5.8.7 Pulse Period ([:SOURce]:PULM:PERiod)

| | |
|-----------------|---|
| Command | [[:SOURce]:PULM:PERiod <value> |
| Format | [[:SOURce]:PULM:PERiod? |
| Instruction | Set the period of the generated pulse when pulse mode is Single or Double. The period determines the repetition frequency of the internal signal Get the period of the generated pulse |
| Parameter Type | Float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | 40 ns ~ 300 s |
| Return | Float, unit: s |
| Default | 10 ms |
| Menu | PULSE > Pulse Period |
| Example | PULM:PER 220 us PULM:PER? |

3.5.8.8 Pulse Period ([:SOURce]:PULM:INT[1]:PERiod)

| | |
|-----------------|---|
| Command | [[:SOURce]:PULM:INT[1]:PERiod <value> |
| Format | [[:SOURce]:PULM:INT[1]:PERiod? |
| Instruction | Set the period of the generated pulse when pulse mode is Single or Double. The period determines the repetition frequency of the internal signal Get the period of the generated pulse |
| Parameter Type | Float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | 40 ns ~ 300 s |
| Return | Float, unit: s |
| Default | 10 ms |
| Menu | PULSE > Pulse Period |

| | |
|---------|--|
| Example | :PULM:INT1:PERiod 900 ns :PULM:INT1:PERiod? |
|---------|--|

3.5.8.9 Pulse Width ([[:SOURce]:PULM:WIDTh])

| | |
|-----------------|--|
| Command | [[:SOURce]:PULM:WIDTh <value> |
| Format | [[:SOURce]:PULM:WIDTh? |
| Instruction | Set the width of the generated pulse when pulse mode is Single or Double. Get the width of the generated pulse when pulse mode is Single or Double. |
| Parameter Type | Float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | 20 ns ~ 300 s |
| Return | Float, unit: s |
| Default | 2 ms |
| Menu | PULSE > Pulse Width |
| Example | PULM:WIDTh 33 us PULM:WIDTh? |

3.5.8.10 Pulse Width ([[:SOURce]:PULM:INT[1]:PWIDTh)

| | |
|-----------------|--|
| Command | [[:SOURce]:PULM:INT[1]:PWIDTh <value> |
| Format | [[:SOURce]:PULM:INT[1]:PWIDTh? |
| Instruction | Set the width of the generated pulse when pulse mode is Single or Double. Get the width of the generated pulse when pulse mode is Single or Double. |
| Parameter Type | Float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | 20 ns ~ 300 s |
| Return | Float, unit: s |
| Default | 2 ms |
| Menu | PULSE > Pulse Width |
| Example | :PULM:INT:PWIDTh 400 ns :PULM:INT:PWIDTh? |

3.5.8.11 Double Pulse Delay ([[:SOURce]:PULM:DOUBle:DELaY])

| | |
|-------------|--|
| Command | [[:SOURce]:PULM:DOUBle:DELaY <value> |
| Format | [[:SOURce]:PULM:DOUBle:DELaY? |
| Instruction | Set the delay from the start of the first pulse to the start of the second pulse when pulse mode is Double. Get the delay from the start of the first pulse to the start of the second pulse when pulse mode is Double. |
| Parameter | Float, unit: ns, us, ms, s, and default is s. |

| | |
|-----------------|--|
| Type | |
| Parameter Range | 20 ns ~ 300 s |
| Return | Float, unit: s |
| Default | 4 ms |
| Menu | PULSE > Double Pulse Delay |
| Example | :PULM:DOUBle:DELaY 2 ms :PULM:DOUBle:DELaY? |

3.5.8.12 #2 Width ([:SOURce]:PULM:DOUBle:WIDTh)

| | |
|-----------------|--|
| Command Format | :[:SOURce]:PULM:DOUBle:WIDTh <time> [:SOURce]:PULM:DOUBle:WIDTh? |
| Instruction | Set the width of the second pulse in the case of double pulse generation Get the width of the second pulse in the case of double pulse generation |
| Parameter Type | Float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | 20 ns ~ 300 s |
| Return | Float, unit: s |
| Default | 2 ms |
| Menu | PULSE > #2 Width |
| Example | PULM:DOUBle:WIDTh 2 s PULM:DOUBle:WIDTh? |

3.5.8.13 Pulse Train Add Row ([:SOURce]:PULM:TRAI:n:PAIR)

| | |
|-----------------|--|
| Command Format | :[:SOURce]:PULM:TRAI:n:PAIR <row> |
| Instruction | Copy and paste the specified row in front of the specified row. |
| Parameter Type | Integer |
| Parameter Range | 1 ~ N, which is the specified row number. N: the length of the current pulse train. |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train > Add |
| Example | PULM:TRAI:n:PAIR 1 |

3.5.8.14 Pulse Train Delete ([:SOURce]:PULM:TRAI:n:DELeTe)

| | |
|----------------|-------------------------------------|
| Command Format | :[:SOURce]:PULM:TRAI:n:DELeTe <row> |
| Instruction | Delete the train pair |
| Parameter | Integer |

| | |
|-----------|---|
| Type | |
| Parameter | 1 ~ N, which is the specified row number. |
| Range | N: the length of the current pulse train. |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train > Delete |
| Example | PULM:TRAI:DELeTe 5 |

3.5.8.15 Pulse Train Edit On Time **([:SOURce]:PULM:TRAI:DATA:ONTIme)**

| | |
|-----------------|---|
| Command Format | [:SOURce]:PULM:TRAI:DATA:ONTIme <raw>,<on_time> |
| Instruction | Edit train ontime value |
| Parameter Type | raw: integer, on_time: float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | raw: 1 ~ count of pair, on_time: 20ns ~ 300s. |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train |
| Example | :PULM:TRAI:DATA:ONTIme 1,10 ms |

3.5.8.16 Pulse Train Edit Off Time **([:SOURce]:PULM:TRAI:DATA:OFFTime)**

| | |
|-----------------|--|
| Command Format | [:SOURce]:PULM:TRAI:DATA:OFFTime <raw>,<off_time> |
| Instruction | Edit train off time value |
| Parameter Type | raw: integer, off_time: float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | raw: 1 ~ count of pair, off_time: 20ns ~ 300s. |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train |
| Example | :PULM:TRAI:DATA:OFFTime 1, 20 ms |

3.5.8.17 Pulse Train Edit Count **([:SOURce]:PULM:TRAI:DATA:COUNt)**

| | |
|----------------|--|
| Command Format | [:SOURce]:PULM:TRAI:DATA:COUNt <raw>,<count> |
| Instruction | Edit train count value |
| Parameter Type | Integer, Integer |

| | |
|-----------|---------------------------|
| Parameter | Raw: 1 ~ count of pair. |
| Range | Count: 1 ~ 65535 |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train |
| Example | :PULM:TRAI:DATA:COUNT 1,3 |

3.5.8.18 Pulse Train Edit ([:SOURce]:PULM:TRAI:CHANGE)

| | |
|-----------------|---|
| Command Format | [:SOURce]:PULM:TRAI:CHANGE <row>,<on_time>,<off_time>,<count> |
| Instruction | Edit train pair value |
| Parameter Type | raw: integer, on_time: float, unit: ns, us, ms, s, and default is s, off_time: float, unit: ns, us, ms, s, and default is s, count: integer. |
| Parameter Range | raw: 1 ~ count of pair, on_time: 20ns ~ 300s, off_time: 20ns ~ 300s, count: 1 ~ 65535. |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train |
| Example | :PULM:TRAI:CHANGE 2,3 ms,500 ns,4 |

3.5.8.19 List Pulse Train ([:SOURce]:PULM:TRAI:LIST?)

| | |
|-----------------|--|
| Command Format | [:SOURce]:PULM:TRAI:LIST? <begin_row>,<end_row> |
| Instruction | View starting row to end row data |
| Parameter Type | Integer, Integer |
| Parameter Range | Begin_row: 1 ~ the count of the pulse list End_row: Begin_row ~ the count of the pulse list |
| Return | String |
| Default | None |
| Menu | PULSE > Pulse Train |
| Example | :PULM:TRAI:LIST? 1,3 |

3.5.8.20 Pulse Train Count ([:SOURce]:PULM:TRAI:COUNt?)

| | |
|----------------|----------------------------|
| Command Format | [:SOURce]:PULM:TRAI:COUNt? |
| Instruction | Get count of train list |
| Parameter Type | None |
| Parameter | None |

SIGLENT

| | |
|---------|---------------------|
| Range | |
| Return | Integer |
| Default | 1 |
| Menu | PULSE > Pulse Train |
| Example | :PULM:TRAI:COUNT? |

3.5.8.21 Pulse Train Clear ([:SOURce]:PULM:TRAI:CLEAr)

| | |
|-----------------|-----------------------------|
| Command Format | [:SOURce]:PULM:TRAI:CLEAr |
| Instruction | Clear train pair list |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train > Store |
| Example | PULM:TRAI:CLEAr |

3.5.8.22 Pulse Train Load ([:SOURce]:PULM:TRAI:LOAD)

| | |
|-----------------|---------------------------------------|
| Command Format | [:SOURce]:PULM:TRAI:LOAD <file> |
| Instruction | Load train pair list |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train > Load |
| Example | PULM:TRAI:LOAD "U-disk3/test.pulstrn" |

3.5.8.23 Pulse Train Store ([:SOURce]:PULM:TRAI:STORE)

| | |
|-----------------|----------------------------------|
| Command Format | [:SOURce]:PULM:TRAI:STORE <file> |
| Instruction | Store train pair list |
| Parameter Type | String |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | PULSE > Pulse Train > Save |
| Example | PULM:TRAI:STORE "test.pulstrn" |

| | |
|--|--|
| | PULM:TRAIIn:STORE "U-disk1/test.pulstrn" |
|--|--|

3.5.8.24 Trigger Out ([:SOURce]:PULM:TRIGger:STaTe)

| | |
|-----------------|--|
| Command | [[:SOURce]:PULM:TRIGger:STaTe ON OFF 1 0 |
| Format | [[:SOURce]:PULM:TRIGger:STaTe? |
| Instruction | Set the trigger output status Get the trigger output status |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 1 |
| Menu | PULSE > Trigger Out |
| Example | PULM:TRIGger:STaTe ON :PULM:TRIGger:STaTe? |

3.5.8.25 Pulse Trigger ([:SOURce]:PULM:TRIGger:MODE)

| | |
|-----------------|---|
| Command | [[:SOURce]:PULM:TRIGger:MODE AUTO KEY EXTeRnal EGATe |
| Format | [[:SOURce]:PULM:TRIGger:MODE? |
| Instruction | Select the trigger mode for pulse modulation Get the trigger mode for pulse modulation |
| Parameter Type | Enumeration |
| Parameter Range | AUTO KEY EXTeRnal EGATe |
| Return | Enumeration |
| Default | AUTO |
| Menu | PULSE > Pulse Trigger |
| Example | PULM:TRIG:MODE EXTeRnal :PULM:TRIGger:MODE? |

3.5.8.26 Trig Polarity ([:SOURce]:PULM:TRIGger:EXTeRnal:GATE:POLarity)

| | |
|-----------------|---|
| Command | [[:SOURce]:PULM:TRIGger:EXTeRnal:GATE:POLarity NORMal INVeRted |
| Format | [[:SOURce]:PULM:TRIGger:EXTeRnal:GATE:POLarity? |
| Instruction | Select the polarity of the gate signal Get the polarity of the gate signal |
| Parameter Type | Enumeration |
| Parameter Range | NORMal INVeRted |
| Return | Enumeration |
| Default | NORMal |

SIGLENT

| | |
|---------|--|
| Menu | PULSE > Pulse Polarity |
| Example | PULM:TRIG:EXT:GATE:POL NORMal :PULM:TRIGger:EXTernal:GATE:POLarity? |

3.5.8.27 Trig Delay ([:SOURce]:PULM:DELay)

| | |
|-----------------|---|
| Command | [:SOURce]:PULM:DELay <value> |
| Format | [:SOURce]:PULM:DELay? |
| Instruction | Set the pulse delay Get the pulse delay |
| Parameter Type | Float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | 140 ns ~ 300 s |
| Return | Float, unit: s |
| Default | 140 ns |
| Menu | PULSE > Trig Delay |
| Example | PULM:DEL 30 ms :PULM:DELay? |

3.5.8.28 Trig Slope ([:SOURce]:PULM:TRIGger:EXTernal:SLOPe)

| | |
|-----------------|--|
| Command | [:SOURce]:PULM:TRIGger:EXTernal:SLOPe NEGative POSitive |
| Format | [:SOURce]:PULM:TRIGger:EXTernal:SLOPe? |
| Instruction | Set the polarity of the active slope of an applied trigger at the PULSE EXT connector Get the polarity of the active slope of an applied trigger at the PULSE EXT connector |
| Parameter Type | Enumeration |
| Parameter Range | NEGative POSitive |
| Return | Enumeration |
| Default | POSitive |
| Menu | PULSE > Trig Slope |
| Example | PULM:TRIG:EXT:SLOP NEG PULM:TRIG:EXT:SLOP? |

3.5.9 [:SOURce]:LFOutput Subsystem

3.5.9.1 LF State ([:SOURce]:LFOutput)

| | |
|-------------|--|
| Command | [:SOURce]:LFOutput ON OFF 1 0 |
| Format | [:SOURce]:LFOutput? |
| Instruction | Activate/deactivate the LF output Get the LF output state |
| Parameter | Boolean |

| | |
|-----------------|--------------------------|
| Type | |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | LF > LF State |
| Example | LFOutput ON LFOutput? |

3.5.9.2 LF Level ([:SOURce]:LFOutput:VOLTage)

| | |
|-----------------|--|
| Command | [:SOURce]:LFOutput:VOLTage <volt> |
| Format | [:SOURce]:LFOutput:VOLTage? |
| Instruction | Set the voltage of the LF output signal Get the voltage of the LF output signal |
| Parameter Type | Float, unit: dBm, uVpp, mVpp, Vpp, nW, uW, mW, Default: Vpp |
| Parameter Range | 1 mVpp ~ 3 Vpp |
| Return | Float, unit: Vpp |
| Default | 0.5 Vpp |
| Menu | LF > LF Voltage |
| Example | LFOutput:VOLTage 2 Vpp LFOutput:VOLTage? |

3.5.9.3 LF Offset ([:SOURce]:LFOutput:OFFSEt)

| | |
|-----------------|--|
| Command | [:SOURce]:LFOutput:OFFSEt <volt> |
| Format | [:SOURce]:LFOutput:OFFSEt? |
| Instruction | Set the voltage offset of the LF output signal Get the voltage offset of the LF output signal |
| Parameter Type | Float, unit: dBm, uV, mV, V, nW, uW, mW, Default: V |
| Parameter Range | $ LFOffset \leq \min(2.5V - \frac{1}{2}LEVEL, 2V)$ |
| Return | Float, unit: V |
| Default | 0 V |
| Menu | LF > LF Offset |
| Example | LFOutput:OFFSEt 1 V LFOutput:OFFSEt? |

3.5.9.4 LF Frequency ([:SOURce]:LFOutput:FREQuency)

| | |
|-------------|--|
| Command | [:SOURce]:LFOutput:FREQuency <freq> |
| Format | [:SOURce]:LFOutput:FREQuency? |
| Instruction | Set LF out put frequency. Get LF out put frequency. |

| | |
|-----------------|--|
| | If the signal source is set to “Internal”, the instrument performs the analog modulations (AM/FM /PM) with this frequency. |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | 0.01 Hz ~ 1 MHz |
| Return | Float, unit: Hz |
| Default | 1 kHz |
| Menu | LF > LF Frequency |
| Example | LFOutput:FREQuency 10 kHz LFOutput:FREQuency? |

3.5.9.5 LF Shape ([:SOURce]:LFOutput:SHAPE)

| | |
|-----------------|--|
| Command Format | [:SOURce]:LFOutput:SHAPE SINE SQUare TRIangle SAWTooth DC [:SOURce]:LFOutput:SHAPE? |
| Instruction | Select the shape of the LF signal Get the shape of the LF signal |
| Parameter Type | Enumeration |
| Parameter Range | SINE SQUare TRIangle SAWTooth DC |
| Return | Enumeration |
| Default | SINE |
| Menu | LF > LF Shape |
| Example | LFOutput:SHAPE TRIangle :LFOutput:SHAPE? |

3.5.9.6 LF Phase ([:SOURce]:LFOutput:PHASE)

| | |
|-----------------|--|
| Command Format | [:SOURce]:LFOutput:PHASE <deg> [:SOURce]:LFOutput:PHASE? |
| Instruction | Set the phase of the LF output signal Get the phase of the LF output signal |
| Parameter Type | Float, unit: deg |
| Parameter Range | -360 deg ~ 360 deg |
| Return | Float, unit: deg |
| Default | 0 deg |
| Menu | LF > LF Phase |
| Example | LFOutput:PHASE 20 LFOutput:PHASE? |

3.5.10 [:SOURce]:LFOutput:SWEep Subsystem

3.5.10.1 Sweep State ([:SOURce]:LFOutput:SWEep)

| | |
|-----------------|---|
| Command | [:SOURce]:LFOutput:SWEep ON OFF 1 0 |
| Format | [:SOURce]:LFOutput:SWEep? |
| Instruction | Activate/Deactivate the LF frequency sweep signal generation Get the state of LF frequency sweep |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | LF Sweep > LF State |
| Example | :LFOutput:SWEep 1 :LFOutput:SWEep? |

3.5.10.2 Sweep Direction ([:SOURce]:LFOutput:SWEep:DIRect)

| | |
|-----------------|--|
| Command | [:SOURce]:LFOutput:SWEep:DIRect UP DOWN |
| Format | [:SOURce]:LFOutput:SWEep:DIRect? |
| Instruction | Set the sweep direction Get the sweep direction |
| Parameter Type | Enumeration |
| Parameter Range | UP DOWN |
| Return | Enumeration |
| Default | UP |
| Menu | LF Sweep > Sweep Direction |
| Example | :LFOutput:SWEep:DIRect DOWN :LFOutput:SWEep:DIRect? |

3.5.10.3 Start Freq ([:SOURce]:LFOutput:SWEep:STArt:FREquency)

| | |
|-----------------|--|
| Command | [:SOURce]:LFOutput:SWEep:STArt:FREquency <freq> |
| Format | [:SOURce]:LFOutput:SWEep:STArt:FREquency? |
| Instruction | Set the start frequency of sweep mode Get the start frequency of sweep mode |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default "Hz" |
| Parameter Range | 0.01 Hz ~ Stop frequency |
| Return | Float, unit: Hz |
| Default | 500 Hz |
| Menu | LF Sweep > Start Freq |

| | |
|---------|---|
| Example | :LFOutput:SWEep:STARt:FREQuency 100 :LFOutput:SWEep:STARt:FREQuency? |
|---------|---|

3.5.10.4 Stop Freq ([:SOURce]:LFOutput:SWEep:STOP:FREQuency)

| | |
|-----------------|--|
| Command | :[:SOURce]:LFOutput:SWEep:STOP:FREQuency <freq> |
| Format | :[:SOURce]:LFOutput:SWEep:STOP:FREQuency? |
| Instruction | Set the stop frequency of sweep mode Get the stop frequency of sweep mode |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | Start frequency ~ Maximum frequency of LF |
| Return | Float, unit: Hz |
| Default | 1.5 kHz |
| Menu | LF Sweep > Stop Freq |
| Example | :LFOutput:SWEep:STOP:FREQuency 1000 :LFOutput:SWEep:STOP:FREQuency? |

3.5.10.5 Center Freq ([:SOURce]:LFOutput:SWEep:CENTER:FREQuency)

| | |
|-----------------|--|
| Command | :[:SOURce]:LFOutput:SWEep:CENTER:FREQuency <freq> |
| Format | :[:SOURce]:LFOutput:SWEep:CENTER:FREQuency? |
| Instruction | Set the center frequency of sweep mode Get the center frequency of sweep mode |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | 0.01 Hz ~ Maximum frequency of LF |
| Return | Float, unit: Hz |
| Default | 1 kHz |
| Menu | LF Sweep > Center Freq |
| Example | :LFOutput:SWEep:CENTER:FREQuency 550 :LFOutput:SWEep:CENTER:FREQuency? |

3.5.10.6 Freq Span ([:SOURce]:LFOutput:SWEep:SPAN:FREQuency)

| | |
|-----------------|--|
| Command | :[:SOURce]:LFOutput:SWEep:SPAN:FREQuency <freq> |
| Format | :[:SOURce]:LFOutput:SWEep:SPAN:FREQuency? |
| Instruction | Set the center frequency of sweep mode Get the center frequency of sweep mode |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default “Hz” |
| Parameter Range | 0 Hz ~ Maximum frequency of LF - 0.01 Hz |
| Return | Float, unit: Hz |

| | |
|---------|---|
| Default | 1 kHz |
| Menu | LF Sweep > Freq Span |
| Example | :LFOutput:SWEep:SPAN:FREQuency 550 :LFOutput:SWEep:SPAN:FREQuency? |

3.5.10.7 Sweep Time ([:SOURce]:LFOutput:SWEep:DWEL1)

| | |
|-----------------|--|
| Command | [:SOURce]:LFOutput:SWEep:DWEL1 <time> |
| Format | [:SOURce]:LFOutput:SWEep:DWEL1? |
| Instruction | Set the sweep time of sweep mode Get the sweep time of sweep mode |
| Parameter Type | Float, unit: ns, us, ms, s, and default is s. |
| Parameter Range | 1 ms ~ 500 s |
| Return | Float, unit: s |
| Default | 1 s |
| Menu | LF Sweep > Sweep Time |
| Example | :LFOutput:SWEep:DWEL1 2 s :LFOutput:SWEep:DWEL1? |

3.5.10.8 Trigger Mode ([:SOURce]:LFOutput:SWEep:TRIGger:TYPE)

| | |
|-----------------|---|
| Command | [:SOURce]:LFOutput:SWEep:TRIGger:TYPE AUTO KEY BUS EXT |
| Format | [:SOURce]:LFOutput:SWEep:TRIGger:TYPE? |
| Instruction | Select the LF frequency sweep trigger mode Get the LF frequency sweep trigger mode |
| Parameter Type | Enumeration |
| Parameter Range | AUTO KEY BUS EXT |
| Return | Enumeration |
| Default | AUTO |
| Menu | LF Sweep > Trigger Mode |
| Example | :LFOutput:SWEep:TRIGger:TYPE KEY :LFOutput:SWEep:TRIGger:TYPE? |

3.5.10.9 Trigger Slope ([:SOURce]:LFOutput:SWEep:XPOLar)

| | |
|-----------------|---|
| Command | [:SOURce]:LFOutput:SWEep:XPOLar POS NEG |
| Format | [:SOURce]:LFOutput:SWEep:XPOLar? |
| Instruction | Select the trigger slope of the external trigger signal Get the trigger slope of the external trigger signal |
| Parameter Type | Enumeration |
| Parameter Range | POS NEG |

SIGLENT

| | |
|---------|---|
| Return | Enumeration |
| Default | POS |
| Menu | LF Sweep > Trigger Slope |
| Example | :LFOutput:SWEep:XPOLar POS :LFOutput:SWEep:XPOLar? |

3.5.10.10 Sweep Shape ([:SOURce]:LFOutput:SWEep:SHApe)

| | |
|-----------------|---|
| Command | [:SOURce]:LFOutput:SWEep:SHApe TRIangle SAWTooth |
| Format | |
| Instruction | Select the waveform shape of the sweep signal Get the waveform shape of the sweep signal |
| Parameter Type | Enumeration |
| Parameter Range | TRIangle SAWTooth |
| Return | Enumeration |
| Default | SAWTooth |
| Menu | LF Sweep > Sweep Shape |
| Example | :LFOutput:SWEep:SHApe TRIangle :LFOutput:SWEep:SHApe? |

3.5.10.11 Sweep Space ([:SOURce]:LFOutput:SWEep:SPACing)

| | |
|-----------------|---|
| Command | [:SOURce]:LFOutput:SWEep:SPACing LINear LOGarithmic |
| Format | |
| Instruction | Select the mode for the calculation of the frequency sweep intervals Get the mode for the calculation of the frequency sweep intervals |
| Parameter Type | Enumeration |
| Parameter Range | LINear LOGarithmic |
| Return | Enumeration |
| Default | LINear |
| Menu | LF Sweep > Sweep Space |
| Example | :LFOutput:SWEep:SPACing LOGarithmic :LFOutput:SWEep:SPACing? |

3.6 Sense Subsystem

3.6.1 Sensor Info (:SENSe[:POWer]:TYPE?)

| | |
|-------------|--|
| Command | :SENSe[:POWer]:TYPE? |
| Format | |
| Instruction | Query the type of sensor connected to the POWER SENSOR connector |

| | |
|-----------------|----------------------|
| Parameter Type | None |
| Parameter Range | None |
| Return | String |
| Default | None |
| Menu | SENSOR > Sensor Info |
| Example | SENSe:TYPE? |

3.6.2 Sensor State (:SENSe[:POWer]:STATUs)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:STATUs OFF ON 0 1 |
| Format | :SENSe[:POWer]:STATUs? |
| Instruction | Set the sensor state Get the sensor state |
| Parameter Type | Boolean |
| Parameter Range | OFF ON 0 1 |
| Return | Boolean |
| Default | 0 |
| Menu | SENSOR > Sensor State |
| Example | SENSe:STATUs ON SENSe:STATUs? |

3.6.3 Measurement (:SENSe[:POWer]:VALue?)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:VALue? |
| Format | |
| Instruction | Indicate the current reading of the sensor |
| Parameter Type | None |
| Parameter Range | None |
| Return | Float, unit: dBm |
| Default | None |
| Menu | SENSOR > Measurement |
| Example | SENSe:VALue? |

3.6.4 Statistics State (:SENSe[:POWer]:STATIStics:STATe)

| | |
|----------------|--|
| Command | :SENSe[:POWer]:STATIStics:STATe ON OFF 1 0 |
| Format | :SENSe[:POWer]:STATIStics:STATe? |
| Instruction | Set statistical status of power meter readings Get statistical status of power meter readings |
| Parameter Type | Boolean |

| | |
|-----------------|--|
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | SENSOR > Statistics |
| Example | SENSe:STATIStics:STATe ON SENSe:STATIStics:STATe? |

3.6.5 Statistics Value (:READ[:POWer]?)

| | |
|-----------------|--|
| Command Format | :READ[:POWer]? |
| Instruction | Indicate the measured mean value and maximum value of power meter readings |
| Parameter Type | None |
| Parameter Range | None |
| Return | String |
| Default | None |
| Menu | SENSOR > Statistics |
| Example | READ? |

3.6.6 Statistics Max Value (:SENSe[:POWer]:STATIStics:MAX?)

| | |
|-----------------|---|
| Command Format | :SENSe[:POWer]:STATIStics:MAX? |
| Instruction | Indicate the measured maximum value of power meter readings |
| Parameter Type | None |
| Parameter Range | None |
| Return | Float, unit: dBm |
| Default | None |
| Menu | SENSOR > Statistics |
| Example | SENSe:STATIStics:MAX? |

3.6.7 Statistics Min Value (:SENSe[:POWer]:STATIStics:MIN?)

| | |
|-----------------|---|
| Command Format | :SENSe[:POWer]:STATIStics:MIN? |
| Instruction | Indicate the measured minimum value of power meter readings |
| Parameter Type | None |
| Parameter Range | None |

| | |
|---------|-----------------------|
| Return | Float, unit: dBm |
| Default | None |
| Menu | SENSOR > Statistics |
| Example | SENSe:STATIStics:MIN? |

3.6.8 Statistics Mean Value (:SENSe[:POWer]:STATIStics:AVG?)

| | |
|-----------------|--|
| Command Format | :SENSe[:POWer]:STATIStics:AVG? |
| Instruction | Indicate the measured mean value of power meter readings |
| Parameter Type | None |
| Parameter Range | None |
| Return | Float, unit: dBm |
| Default | None |
| Menu | SENSOR > Statistics |
| Example | SENSe:STATIStics:AVG? |

3.6.9 Statistics Count (:SENSe[:POWer]:STATIStics:COUNt?)

| | |
|-----------------|--|
| Command Format | :SENSe[:POWer]:STATIStics:COUNt? |
| Instruction | Indicate the number of measurements being used to calculate the statistics |
| Parameter Type | None |
| Parameter Range | None |
| Return | Integer |
| Default | None |
| Menu | SENSOR > Statistics |
| Example | SENSe:STATIStics:COUNt? |

3.6.10 Statistics Clear (:SENSe[:POWer]:STATIStics:CLEAr)

| | |
|-----------------|---|
| Command Format | :SENSe[:POWer]:STATIStics:CLEAr |
| Instruction | Clear the statistics counter of power meter |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |

| | |
|---------|------------------------|
| Default | None |
| Menu | SENSOR > Statistics |
| Example | SENSe:STATISTICS:CLEAr |

3.6.11 Auto Zero (:CALibration:ZERO:TYPE)

| | |
|-----------------|---|
| Command | :CALibration:ZERO:TYPE INTernal EXTernal |
| Format | :CALibration:ZERO:TYPE? |
| Instruction | Select zero type of power meter Get zero type of power meter |
| Parameter Type | Enumeration |
| Parameter Range | INTernal EXTernal |
| Return | Enumeration |
| Default | INTernal |
| Menu | SENSOR > Auto Zero |
| Example | CALibration:ZERO:TYPE EXTernal :CALibration:ZERO:TYPE? |

3.6.12 Zeroing (:SENSe[:POWER]:ZERO)

| | |
|-----------------|-----------------------------------|
| Command | :SENSe[:POWER]:ZERO |
| Format | |
| Instruction | Perform zeroing of the sensor |
| Parameter Type | None |
| Parameter Range | None |
| Return | None |
| Default | None |
| Menu | SENSOR > Click to perform zeroing |
| Example | :SENSe:ZERO |

3.6.13 Frequency Type (:SENSe[:POWER]:SOURce)

| | |
|-----------------|---|
| Command | :SENSe[:POWER]:SOURce RF USER |
| Format | :SENSe[:POWER]:SOURce? |
| Instruction | Select the signal source for the measurement Get the signal source for the measurement |
| Parameter Type | Enumeration |
| Parameter Range | RF USER |
| Return | Enumeration |
| Default | RF |
| Menu | SENSOR > Frequency |
| Example | SENSe:SOURce RF SENSe:SOURce? |

3.6.14 Frequency (:SENSe[:POWer]:FREQuency)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:FREQuency <freq> |
| Format | :SENSe[:POWer]:FREQuency? |
| Instruction | Set the frequency for frequency type "USER" Get the frequency for frequency type "USER" |
| Parameter Type | Float, unit: Hz, kHz, MHz, GHz, Default "Hz" |
| Parameter Range | 9 kHz ~ Full frequency range |
| Return | Float, unit: Hz |
| Default | None |
| Menu | SENSOR > Frequency |
| Example | SENSe:FREQuency 1 MHz SENSe:FREQuency? |

3.6.15 Level Offset State (:SENSe[:POWer]:OFFSet:STATe)

| | |
|-----------------|---|
| Command | :SENSe[:POWer]:OFFSet:STATe ON OFF 1 0 |
| Format | :SENSe[:POWer]:OFFSet:STATe? |
| Instruction | Switch the power offset switch status Get the power offset switch status |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | SENSOR > Level Offset |
| Example | SENSe:OFFSet:STATe ON SENSe:OFFSet:STATe? |

3.6.16 Level Offset (:SENSe[:POWer]:OFFSet)

| | |
|-----------------|---|
| Command | :SENSe[:POWer]:OFFSet <power> |
| Format | :SENSe[:POWer]:OFFSet? |
| Instruction | The command enters a level offset which is mathematically added to the measured level value Get the level offset which is mathematically added to the measured level value |
| Parameter Type | Float, unit: dB |
| Parameter Range | Limit by power sensor. |
| Return | Float, unit: dB |
| Default | 0 dB |
| Menu | SENSOR > Level Offset |

| | |
|---------|----------------------------------|
| Example | SENSe:OFFSet 10 SENSe:OFFSet? |
|---------|----------------------------------|

3.6.17 Average Type (:SENSe[:POWer]:FILTer:TYPE)

| | |
|-----------------|---|
| Command | :SENSe[:POWer]:FILTer:TYPE AUTO USER NSRatio |
| Format | :SENSe[:POWer]:FILTer:TYPE? |
| Instruction | Select the averaging mode Get the averaging mode |
| Parameter Type | Enumeration |
| Parameter Range | AUTO USER NSRatio |
| Return | Enumeration |
| Default | AUTO |
| Menu | SENSOR > Averaging |
| Example | SENSe:FILTer:TYPE USER SENSe:FILTer:TYPE? |

3.6.18 Average Times (:SENSe[:POWer]:FILTer:LENGth)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:FILTer:LENGth <length> |
| Format | :SENSe[:POWer]:FILTer:LENGth? |
| Instruction | Set the average number times |
| Parameter Type | Integer |
| Parameter Range | Limit by power sensor |
| Return | Float |
| Default | None |
| Menu | SENSOR > Averaging |
| Example | SENSe:FILTer:LENGth 10 SENSe:FILTer:LENGth? |

3.6.19 Internal Noise (:SENSe[:POWer]:FILTer:NSRatio)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:FILTer:NSRatio <noise> |
| Format | :SENSe[:POWer]:FILTer:NSRatio? |
| Instruction | The power sensor will control the internal noise that does not exceed the set value of the fixed noise parameter |
| Parameter Type | Float, unit: dB |
| Parameter Range | Limit by power sensor. |
| Return | Float, unit: dB |
| Default | None |
| Menu | SENSOR > Averaging |

| | |
|---------|---|
| Example | SENSe:FILTer:NSRatio 1 SENSe:FILTer:NSRatio? |
|---------|---|

3.6.20 Logging (:SENSe[:POWer]:LOGGing:STATe)

| | |
|-----------------|--|
| Command | :SENSe[:POWer]:LOGGing:STATe <state> |
| Format | :SENSe[:POWer]:LOGGing:STATe? |
| Instruction | Set logging state Get logging state |
| Parameter Type | Boolean |
| Parameter Range | ON OFF 1 0 |
| Return | Boolean |
| Default | 0 |
| Menu | SENSOR > Logging |
| Example | SENSe:LOGGing:STATe ON SENSe:LOGGing:STATe? |

4. Programming Examples

This chapter gives some examples for the programmer. In these examples you can see how to use the VISA or sockets, in combination with the commands have been described above to control the signal generator. By following these examples, you can develop many more applications.

4.1 VISA Examples

4.1.1 VC++ Example

Environment: Win10 64bit system, Visual Studio

The functions of this example: Use National Instruments NI-VISA to control the device with USBTMC or TCP/IP access and perform write and read operations.

Follow the steps to finish the example:

1. Open Visual Studio, create a new VC++ win32 console project.
2. Set the project environment to use the NI-VISA lib, there are two ways to use NI-VISA, static or automatic:

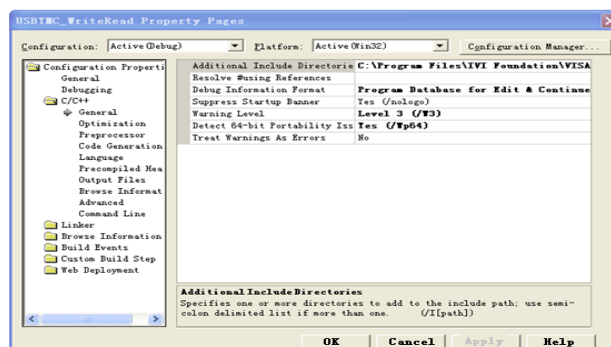
(1) Static: find files: visa.h, visatype.h, visa32.lib in NI-VISA install path. Copy them to your project, and add them into project. In the projectname.cpp file, add the follow two lines:

```
#include "visa.h"
```

```
#pragma comment(lib, "visa32.lib")
```

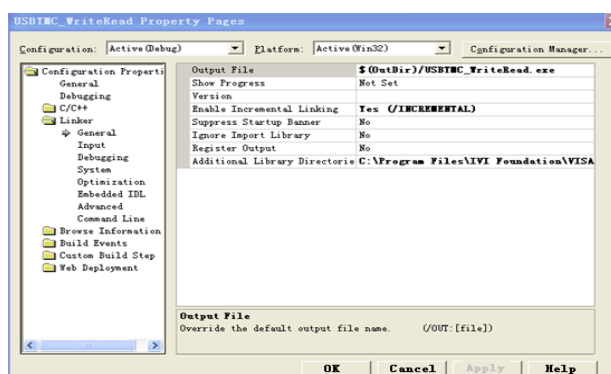
(2) Automatic:

Set the .h file include directory, the NI-VISA install path, in our computer we set the path is: C:\Program Files\IVI Foundation\VISA\WinNT\include. Set this path to project---properties---c/c++---General---Additional Include Directories: See the picture:

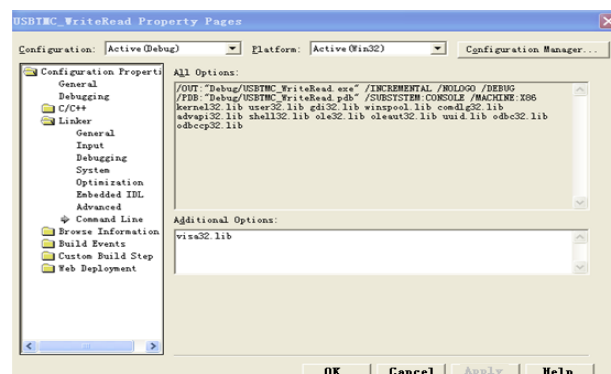


Set lib path set lib file:

Set lib path: the NI-VISA install path, in our computer we set the path is: C:\Program Files\IVI Foundation\VISA\WinNT\lib\msc. Set this path to project---properties---Linker---General---Additional Library Directories: as seen in the pictures below.



Set lib file: project---properties---Linker---Command Line---Additional Options: visa32.lib



Include visa.h file: In the projectname.cpp file:

```
#include <visa.h>
```

3. Add the following code:

(1) USBTMC access code.

```

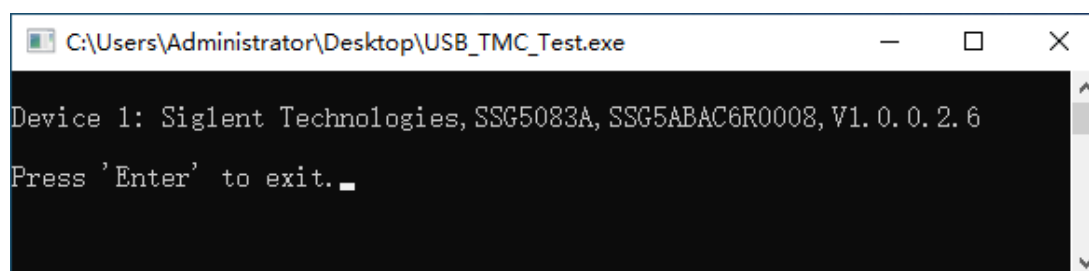
Write a function Usbtmc_test:
int Usbtmc_test()
{
    /* This code demonstrates sending synchronous read & write commands */
    /* to an USB Test & Measurement Class (USBTMC) instrument using */
    /* NI-VISA */
    /* The example writes the "*IDN?\n" string to all the USBTMC */
    /* devices connected to the system and attempts to read back */
    /* results using the write and read functions. */
    /* The general flow of the code is */
    /* Open Resource Manager */
    /* Open VISA Session to an Instrument */
    /* Write the Identification Query Using viPrintf */
    /* Try to Read a Response With viScanf */
    /* Close the VISA Session */
    /***/
    ViSession defaultRM;
    ViSession instr;
    ViUInt32 numInstrs;
    ViFindList findList;
    ViStatus status;
    char instrResourceString[VI_FIND_BUFLEN];
    unsigned char buffer[100];
    int i;
    /** First we must call viOpenDefaultRM to get the manager
    * handle. We will store this handle in defaultRM.*/
    status = viOpenDefaultRM (&defaultRM);
    if (status<VI_SUCCESS)
    {
        printf ("Could not open a session to the VISA Resource Manager!\n");
        return status;
    }
    /* Find all the USB TMC VISA resources in our system and store the number of resources in the
    system in numInstrs.*/
    status = viFindRsrc (defaultRM, "USB?*INSTR", &findList, &numInstrs, instrResourceString);
    if (status<VI_SUCCESS)
    {
        printf ("An error occurred while finding resources.\nPress 'Enter' to continue.");
        fflush(stdin);
        getchar();
        viClose (defaultRM);
        return status;
    }
    /** Now we will open VISA sessions to all USB TMC instruments.
    * We must use the handle from viOpenDefaultRM and we must
    * also use a string that indicates which instrument to open. This
    * is called the instrument descriptor. The format for this string
    * can be found in the function panel by right clicking on the
    * descriptor parameter. After opening a session to the
    * device, we will get a handle to the instrument which we
    * will use in later VISA functions. The AccessMode and Timeout
    * parameters in this function are reserved for future
    * functionality. These two parameters are given the value VI_NULL.*/
    for (i=0; i<int(numInstrs); i++)
    {
        if (i> 0)
        {
            viFindNext (findList, instrResourceString);

```

```
}
status = viOpen (defaultRM, instrResourceString, VI_NULL, VI_NULL, &instr);
if (status<VI_SUCCESS)
{
printf ("Cannot open a session to the device %d.\n", i+1);
continue ;
}
/* * At this point we now have a session open to the USB TMC instrument.
* We will now use the viPrintf function to send the device the string "*IDN?\n",
* asking for the device's identification. */
char * cmmmand ="*IDN?\n";
status = viPrintf (instr, cmmmand);
if (status<VI_SUCCESS)
{
printf ("Error writing to the device %d.\n", i+1);
status = viClose (instr);
continue;
}
/** Now we will attempt to read back a response from the device to
* the identification query that was sent. We will use the viScanf
* function to acquire the data.
* After the data has been read the response is displayed. */
status = viScanf(instr, "%t", buffer);
if (status<VI_SUCCESS)
{
printf ("Error reading a response from the device %d.\n", i+1);
}
else
{
printf ("\nDevice %d: %s\n", i+1, buffer);
}
status = viClose (instr);
}
/** Now we will close the session to the instrument using
* viClose. This operation frees all system resources. */
status = viClose (defaultRM);
printf("Press 'Enter' to exit.");
fflush(stdin);
getchar();
return 0;
}

int _tmain(int argc, _TCHAR* argv[])
{
Usbtmc_test();
return 0;
}
```

The run result:



```
C:\Users\Administrator\Desktop\USB_TMC_Test.exe
Device 1: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
Press 'Enter' to exit.
```

(2) TCP/IP access code.

```

    Write a function TCP_IP_Test:
int TCP_IP_Test(char *pIP)
{
    char outputBuffer[VI_FIND_BUFLEN];
    ViSession defaultRM, instr;
    ViStatus status;

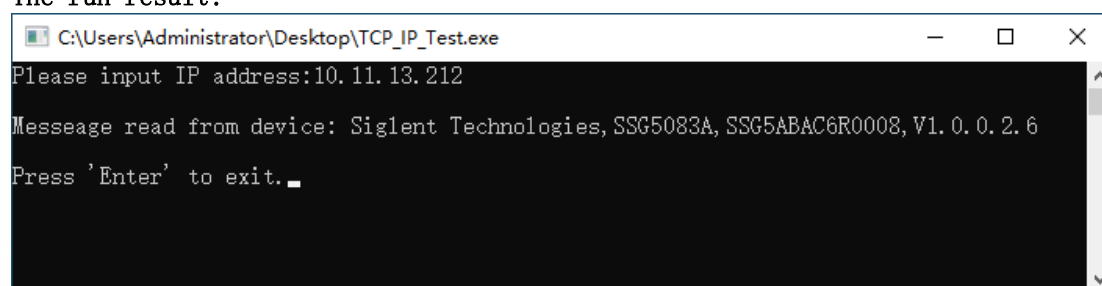
    /* First we will need to open the default resource manager. */
    status = viOpenDefaultRM (&defaultRM);
    if (status<VI_SUCCESS)
    {
        printf("Could not open a session to the VISA Resource Manager!\n");
    }
    /* Now we will open a session via TCP/IP device */
    char head[256] = "TCPIP0::";
    char tail[] = "::INSTR";

    strcat(head,pIP);
    strcat(head,tail);
    status = viOpen (defaultRM, head, VI_LOAD_CONFIG, VI_NULL, &instr);
    if (status<VI_SUCCESS)
    {
        printf ("An error occurred opening the session\n");
        viClose(defaultRM);
    }
    status = viPrintf(instr, "%idn?\n");
    if (status<VI_SUCCESS)
    {
        printf("Error writing to the device.\n");
        viClose(defaultRM);
    }
    status = viScanf(instr, "%t", outputBuffer);
    if (status<VI_SUCCESS)
    {
        printf("viRead failed with error code: %x \n",status);
        viClose(defaultRM);
    }
    else
    {
        printf ("\nMesseage read from device: %s\n", 0,outputBuffer);
    }
    status = viClose (instr);
    status = viClose (defaultRM);
    printf("Press 'Enter' to exit.");
    fflush(stdin);
    getchar();
    return 0;
}

int _tmain(int argc, _TCHAR* argv[])
{
    printf("Please input IP address:");
    char ip[256];
    fflush(stdin);
    gets(ip);
    TCP_IP_Test(ip);
    return 0;
}

```

The run result:



```
C:\Users\Administrator\Desktop\TCP_IP_Test.exe
Please input IP address:10.11.13.212
Message read from device: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
Press 'Enter' to exit.
```

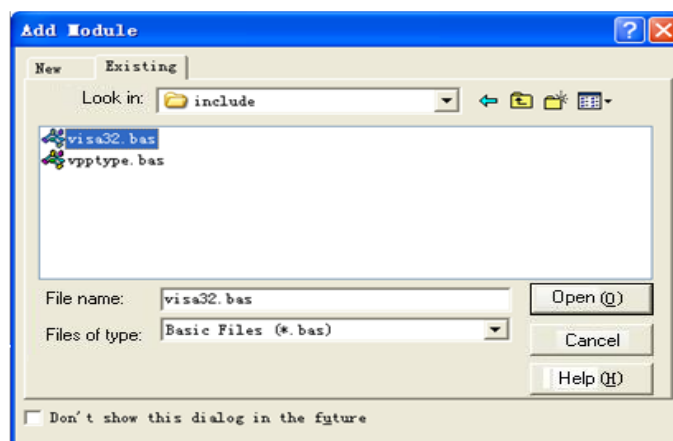
4.1.2 VB Example

Environment: Win10 64bit system, Microsoft Visual Basic 6.0

The function of this example: Use National Instruments NI-VISA to control the device with USBTMC and TCP/IP access and perform write and read operations.

Follow the steps to complete the example:

1. Open Visual Basic, build a standard application program project (Standard EXE)
2. Set the project environment to use the NI-VISA lib, Click the Existing tab of Project>>Add Existing Item. Search for the visa32.bas file in the include folder under the NI-VISA installation path and add the file.



This allows the VISA functions and VISA data types to be used in a program.

3. Add the following code:

(1) USBTMC access code.

Write a function Usbtmc_test:

```
Private Function Usbtmc_test() As Long
    ' This code demonstrates sending synchronous read & write commands
    ' to an USB Test & Measurement Class (USBTMC) instrument using
    ' NI-VISA
    ' The example writes the "*IDN?\n" string to all the USBTMC
    ' devices connected to the system and attempts to read back
```

```

' results using the write and read functions.
' The general flow of the code is
' Open Resource Manager
' Open VISA Session to an Instrument
' Write the Identification Query Using viWrite
' Try to Read a Response With viRead
' Close the VISA Session
Const MAX_CNT = 200

Dim defaultRM As Long
Dim instrsesn As Long
Dim numInstrs As Long
Dim findList As Long
Dim retCount As Long

Dim status As Long
Dim instrResourceString As String * VI_FIND_BUFLEN
Dim Buffer As String * MAX_CNT
Dim i As Integer
' First we must call viOpenDefaultRM to get the manager
' handle. We will store this handle in defaultRM.
status = viOpenDefaultRM(defaultRM)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Could not open a session to the VISA Resource Manager!"
    Usbtmc_test = status
    Exit Function
End If

' Find all the USB TMC VISA resources in our system and store the
' number of resources in the system in numInstrs.
status = viFindRsrc(defaultRM, "USB?*INSTR", findList, numInstrs, instrResourceString)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "An error occurred while finding resources."
    viClose(defaultRM)
    Usbtmc_test = status
    Exit Function
End If

' Now we will open VISA sessions to all USB TMC instruments.
' We must use the handle from viOpenDefaultRM and we must
' also use a string that indicates which instrument to open. This
' is called the instrument descriptor. The format for this string
' can be found in the function panel by right clicking on the
' descriptor parameter. After opening a session to the
' device, we will get a handle to the instrument which we
' will use in later VISA functions. The AccessMode and Timeout
' parameters in this function are reserved for future
' functionality. These two parameters are given the value VI_NULL.
For i = 0 To numInstrs
    If (i > 0) Then
        status = viFindNext(findList, instrResourceString)
    End If
    status = viOpen(defaultRM, instrResourceString, VI_NULL, VI_NULL, instrsesn)
    If (status < VI_SUCCESS) Then
        resultTxt.Text = "Cannot open a session to the device " + CStr(i + 1)
        GoTo NextFind
    End If

    ' At this point we now have a session open to the USB TMC instrument.

```

```
' We will now use the viWrite function to send the device the string "*IDN?",
' asking for the device's identification.
status = viWrite(instrsesn, "*IDN?", 5, retCount)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error writing to the device."
    status = viClose(instrsesn)
    GoTo NextFind
End If

' Now we will attempt to read back a response from the device to
' the identification query that was sent. We will use the viRead
' function to acquire the data.
' After the data has been read the response is displayed.
status = viRead(instrsesn, Buffer, MAX_CNT, retCount)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error reading a response from the device." + CStr(i + 1)
Else
    resultTxt.Text = "Read from device: " + CStr(i + 1) + " " + Buffer
End If
status = viClose(instrsesn)

Next i

' Now we will close the session to the instrument using
' viClose. This operation frees all system resources.
status = viClose(defaultRM)
Usbtmc_test = 0
End Function
```

(2) TCP/IP access code.

Write a function TCP_IP_Test:

```
Private Function TCP_IP_Test(ByVal ip As String) As Long
    Dim outputBuffer As String * VI_FIND_BUFLen
    Dim defaultRM As Long
    Dim instrsesn As Long
    Dim status As Long
    Dim count As Long

    ' First we will need to open the default resource manager.
    status = viOpenDefaultRM(defaultRM)
    If (status < VI_SUCCESS) Then
        resultTxt.Text = "Could not open a session to the VISA Resource Manager!"
        TCP_IP_Test = status
        Exit Function
    End If

    ' Now we will open a session via TCP/IP device
    status = viOpen(defaultRM, "TCPIP0::" + ip + "::INSTR", VI_LOAD_CONFIG, VI_NULL, instrsesn)
    If (status < VI_SUCCESS) Then
        resultTxt.Text = "An error occurred opening the session"
        viClose(defaultRM)
        TCP_IP_Test = status
        Exit Function
    End If

    status = viWrite(instrsesn, "*IDN?", 5, count)
    If (status < VI_SUCCESS) Then
        resultTxt.Text = "Error writing to the device."
```



```

End If
status = viRead(instrsesn, outputBuffer, VI_FIND_BUFLEN, count)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error reading a response from the device." + CStr(i + 1)
Else
    resultTxt.Text = "read from device:" + outputBuffer
End If
status = viClose(instrsesn)
status = viClose(defaultRM)
TCP_IP_Test = 0
End Function

```

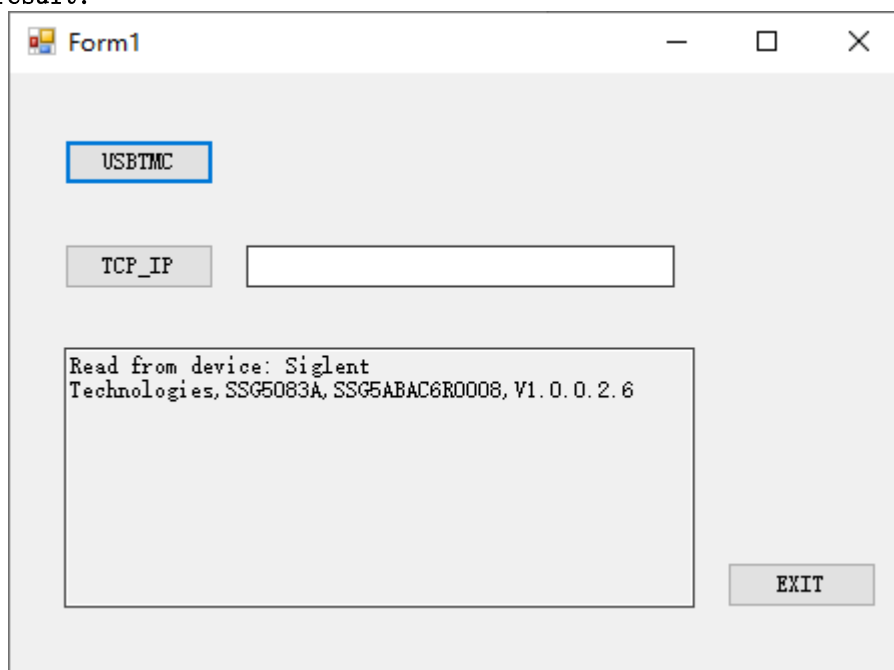
(3)Button control code:

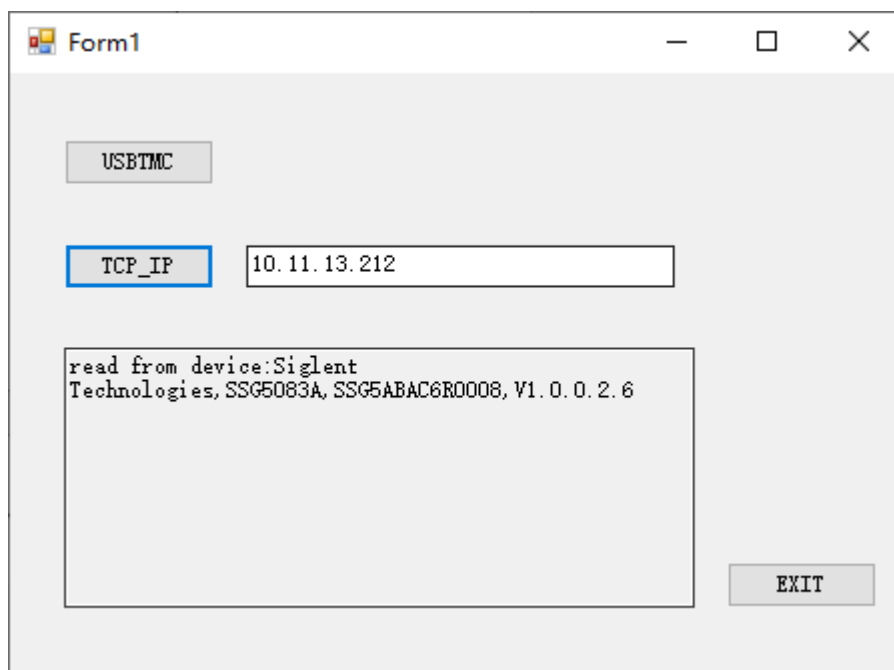
```

Private Sub exitBtn_Click()
    End
End Sub
Private Sub tcpipBtn_Click()
    Dim stat As Long
    stat = TCP_IP_Test(ipTxt.Text)
    If (stat < VI_SUCCESS) Then
        resultTxt.Text = Hex(stat)
    End If
End Sub
Private Sub usbBtn_Click()
    Dim stat As Long
    stat = Usbtmc_test
    If (stat < VI_SUCCESS) Then
        resultTxt.Text = Hex(stat)
    End If
End Sub

```

The run result:





4.1.3 MATLAB Example

Environment: Win10 64bit system, MATLAB R2021a

The function of this example: Use National Instruments NI-VISA to control the device with USBTCM or TCP/IP access and perform write and read operations.

Follow the steps to complete the example:

1. Open MATLAB, modify the **current directory**. In this demo, the current directory is modified to D:\USBTMC_TCPIP_Demo.
2. Click **File>>New>>Script** in the Matlab interface to create an empty M file
3. Add codes:

(1) USBTCM access code
Write a function Usbtmc_test.

```
function Usbtmc_test()  
% This code demonstrates sending synchronous read & write commands  
% to an USB Test & Measurement Class (USBTMC) instrument using  
% NI-VISA  
  
%Create a VISA-USB object connected to a USB instrument  
vu = visa('ni','USB0::0xF4EC::0x1503::SSG5ABAC6R0008::INSTR');  
  
%Open the VISA object created  
fopen(vu);  
  
%Send the string "*IDN?", asking for the device's identification.  
fprintf(vu,'*IDN?');
```

```

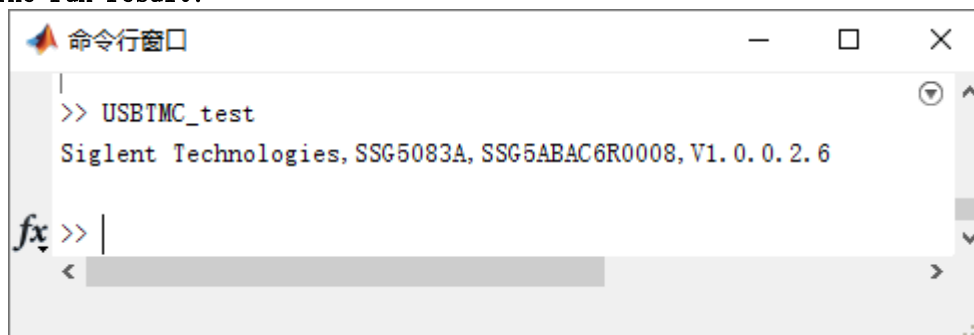
%Request the data
outputbuffer = fscanf(vu);
disp(outputbuffer);

%Close the VISA object
fclose(vu);
delete(vu);
clear vu;

end

```

The run result:



(2) TCP/IP access code.

Write a function TCP_IP_Test:

```

function TCP_IP_test()
% This code demonstrates sending synchronous read & write commands
% to an TCP/IP instrument using NI-VISA

%Create a VISA-TCP/IP object connected to an instrument
%configured with IP address.
vt = visa('ni', ['TCPIP0::', '10.11.13.212', '::INSTR']);

%Open the VISA object created
fopen(vt);

%Send the string "*IDN?", asking for the device's identification.
fprintf(vt, '*IDN?');

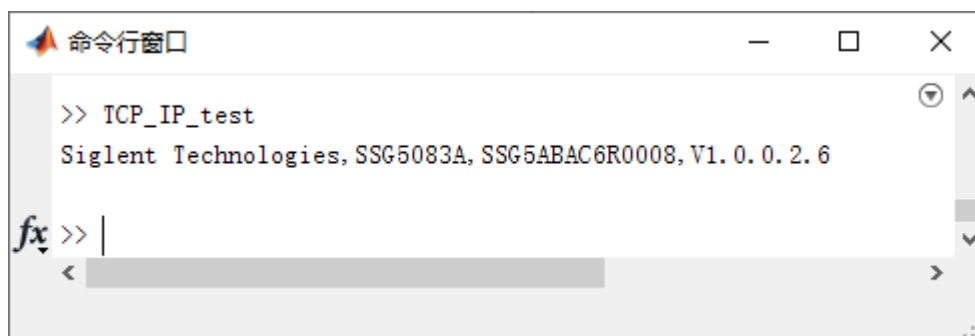
%Request the data
outputbuffer = fscanf(vt);
disp(outputbuffer);

%Close the VISA object
fclose(vt);
delete(vt);
clear vt;

end

```

The run result:



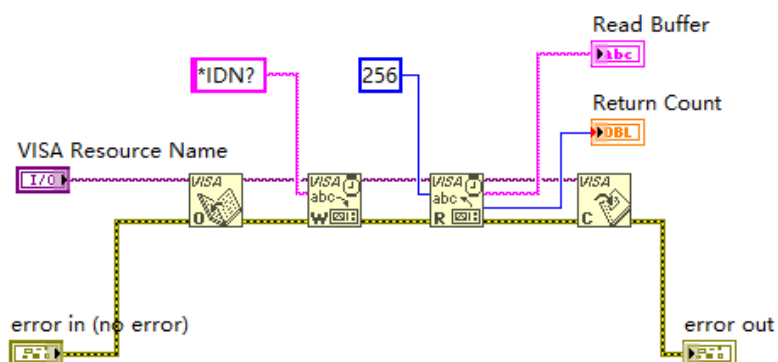
4.1.4 LabVIEW Example

Environment: Win7 32bit system, LabVIEW 2011

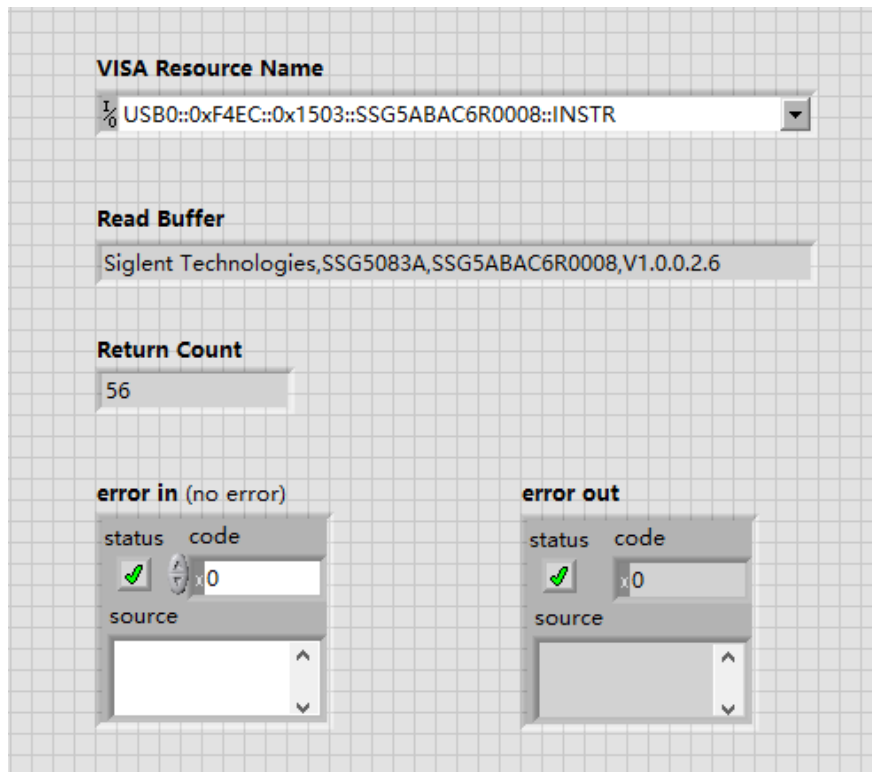
The functions of this example: Use National Instruments NI-VISA to control the device with USBTMC and TCP/IP access to perform write and read operations.

Follow the steps to complete the example:

1. Open LabVIEW, create a VI file.
2. Add controls. Right-click in the **Front Panel** interface, select and add **VISA resource name**, error in, error out and some indicators from the Controls column.
3. Open the **Block Diagram** interface. Right-click on the **VISA resource name** and you can select and add the following functions from VISA Palette from the pop-up menu: **VISA Write**, **VISA Read**, **VISA Open** and **VISA Close**.
4. Connect them as shown in the figure below



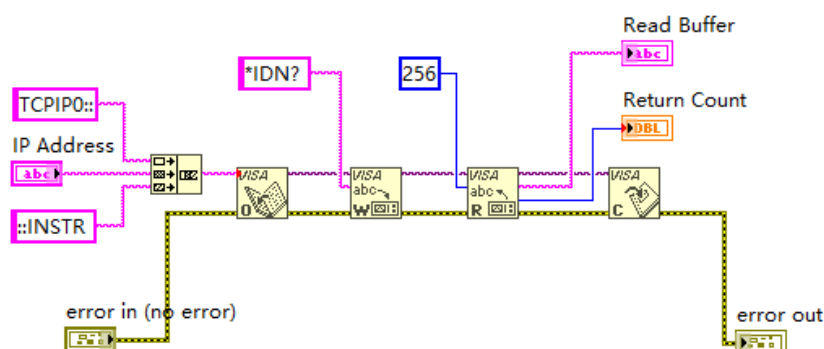
5. Select the device resource from the VISA Resource Name list box and run the program.



In this example, the VI opens a VISA session to a USBTMC device, writes a command to the device, and reads back the response. In this example, the specific command being sent is the device ID query. Check with your device manufacturer for the device command set. After all communication is complete, the VI closes the VISA session.

6. Communicating with the device via TCP/IP is similar to USBTMC. But you need to change VISA Write and VISA Read Function to Synchronous I/O. The LabVIEW default is asynchronous I/O. Right-click the node and select Synchronous I/O Mod>>Synchronous from the shortcut menu to write or read data synchronously.

7. Connect them as shown in the figure below



8. Input the IP address and run the program.

The screenshot shows a software interface with a grid background. It contains several input fields and status indicators:

- IP Address:** A text box containing "10.11.13.212".
- Read Buffer:** A text box containing "Siglent Technologies,SSG5083A,SSG5ABAC6R0008,V1.0.0.2.6".
- Return Count:** A text box containing "56".
- error in (no error):** A panel with a "status" field showing a green checkmark, a "code" field showing "x0", and a "source" field with a scrollable list.
- error out:** A panel with a "status" field showing a green checkmark, a "code" field showing "x0", and a "source" field with a scrollable list.

4.2 Socket Examples

4.2.1 Python Example

Python is an interpreted programming language that lets you work quickly and is very portable. Python has a low-level networking module that provides access to the socket interface. Python scripts can be written for sockets to do a variety of test and measurements tasks.

Environment: Win10 64bit system, Python v3.6.5

The functions of this example: Opens a socket, sends a query, and closes the socket. It does this loop 10 times.

Below is the code of the script:

```
#!/usr/bin/env python
#-*- coding:utf-8 -*-
#-----
# The short script is an example that open a socket, sends a query,
# print the return message and closes the socket.
#-----
import socket # for sockets
import sys # for exit
import time # for sleep
#-----
remote_ip = "10.11.13.32" # should match the instrument' s IP address
port = 5025 # the port number of the instrument service

def SocketConnect():
    try:
        #create an AF_INET, STREAM socket (TCP)
```

```

        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    except socket.error:
        input('Failed to create socket. \nPress "Enter" to exit: ')
        sys.exit()
    try:
        #Connect to remote server
        s.connect((remote_ip , port))
    except socket.error:
        input('Failed to connect to ip %s!\nPress "Enter" to exit: ' % remote_ip)
    s.close()
    sys.exit()
    return s

def SocketQuery(Sock, cmd):
    try :
        #Send cmd string
        Sock.sendall(cmd)
        time.sleep(1)
    except socket.error:
        #Send failed
        input('Send failed!\nPress "Enter" to exit: ')
        SocketClose(Sock)
        sys.exit()
    reply = Sock.recv(4096)
    reply = reply.decode()
    return reply

def SocketClose(Sock):
    #close the socket
    Sock.close()
    time.sleep(.300)

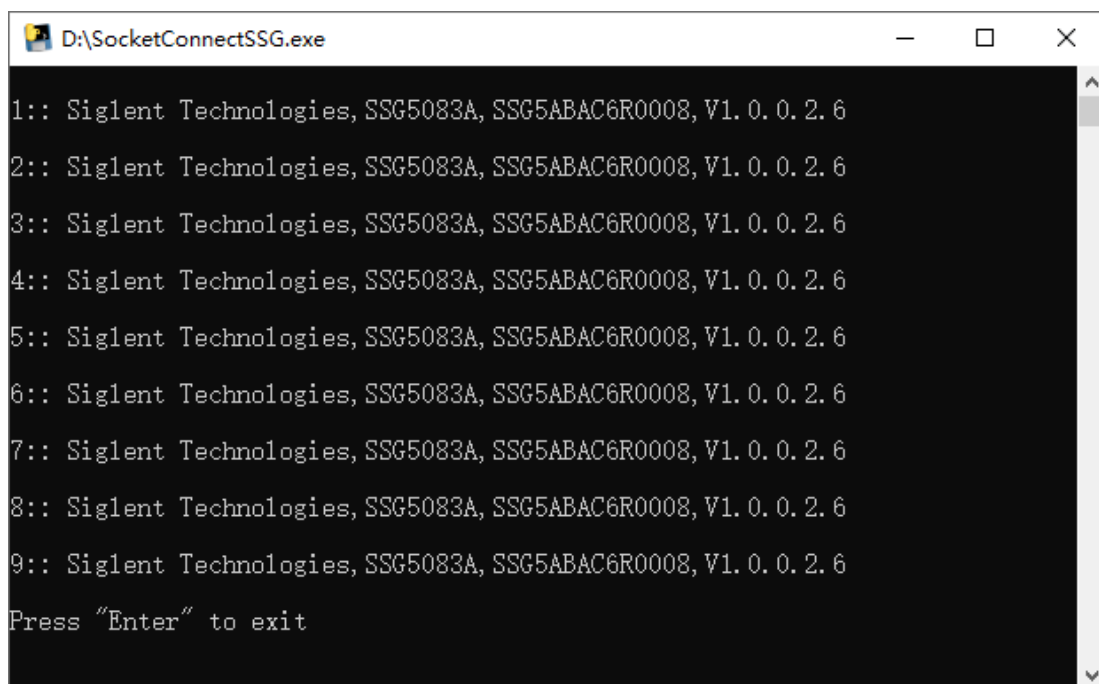
def main():
    # Body: send the SCPI commands *IDN? 10 times and print the return message
    s = SocketConnect()

    count = 0
    for i in range(10):
        qStr = SocketQuery(s, b'*IDN?\n')
        print (str(count) + ">:: " + qStr)
        count = count + 1
    SocketClose(s)
    input('Press "Enter" to exit')

if __name__ == '__main__':
    main()

```

The run result:



```
D:\SocketConnectSSG.exe

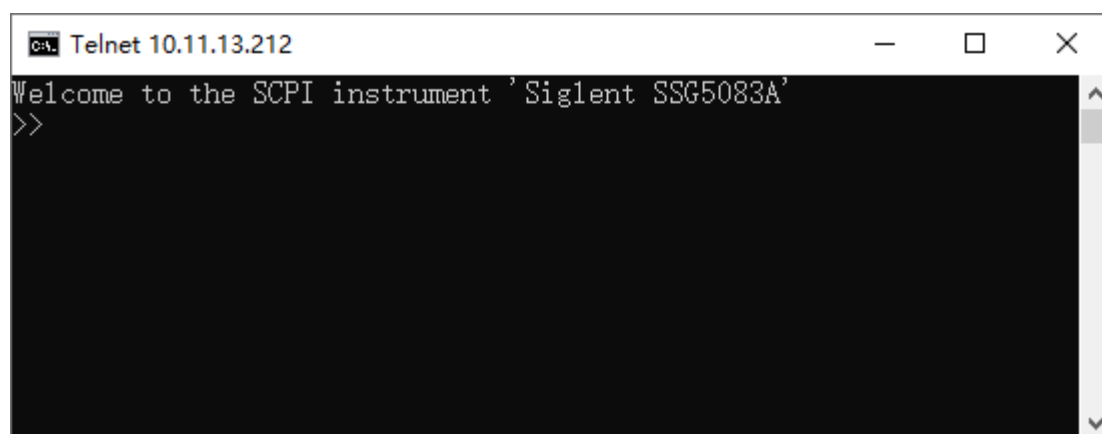
1:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
2:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
3:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
4:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
5:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
6:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
7:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
8:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
9:: Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
Press "Enter" to exit
```

4.2.2 Telnet Example

Telnet SCPI: Provides the ability to send single SCPI commands from a remote PC to the signal generator using LAN port number 5024.

How to send single SCPI commands using Telnet:

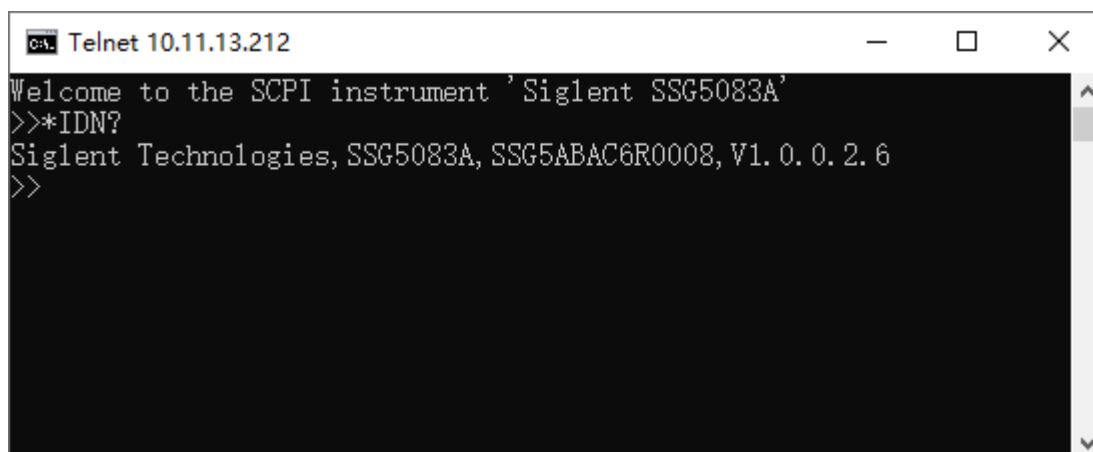
1. On the remote PC, click Start, then Run cmd
2. Type: **telnet <ip address> 5024**
3. A Telnet window with a >> prompt should appear on the remote PC screen.



```
Telnet 10.11.13.212

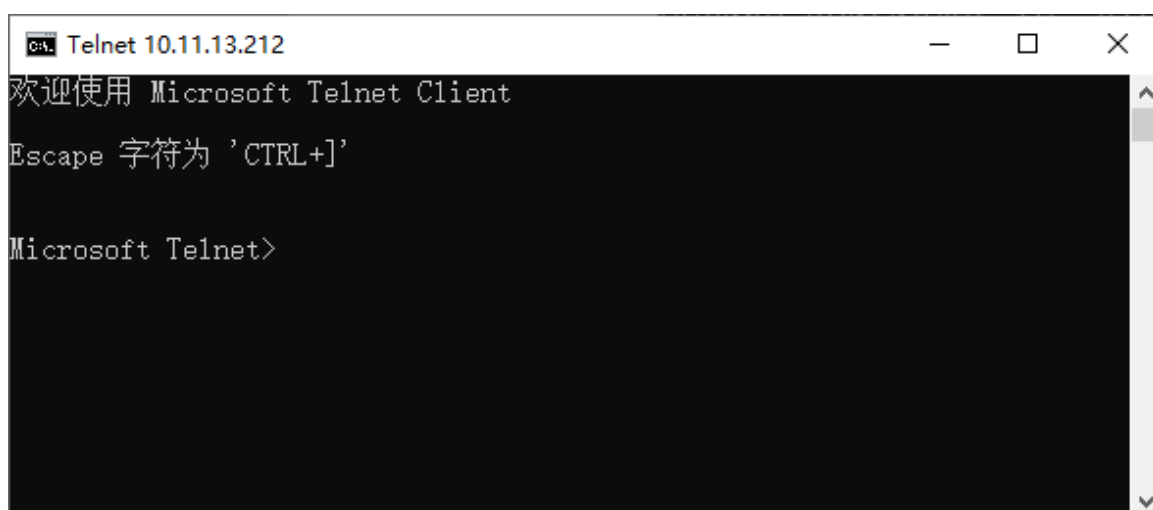
Welcome to the SCPI instrument 'Siglent SSG5083A'
>>
```

4. From the SCPI prompt:
 - Type single SCPI commands. Press Enter to send the command.



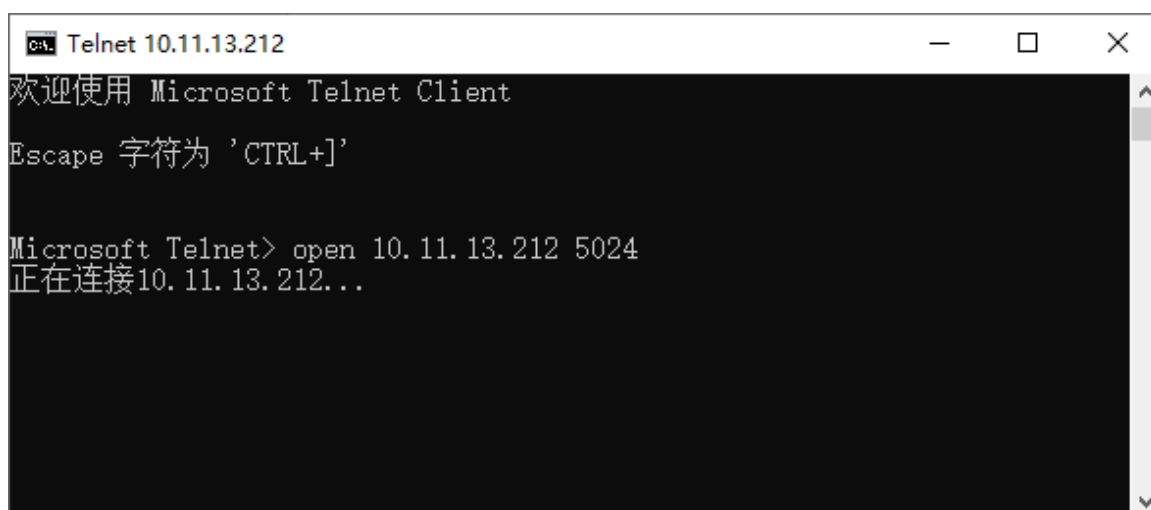
```
CA. Telnet 10.11.13.212
Welcome to the SCPI instrument 'Siglent SSG5083A'
>>*IDN?
Siglent Technologies, SSG5083A, SSG5ABAC6R0008, V1.0.0.2.6
>>
```

- To exit the telnet window click X in the upper-right corner.
- To get a normal telnet prompt, press Ctrl +] (closing bracket).



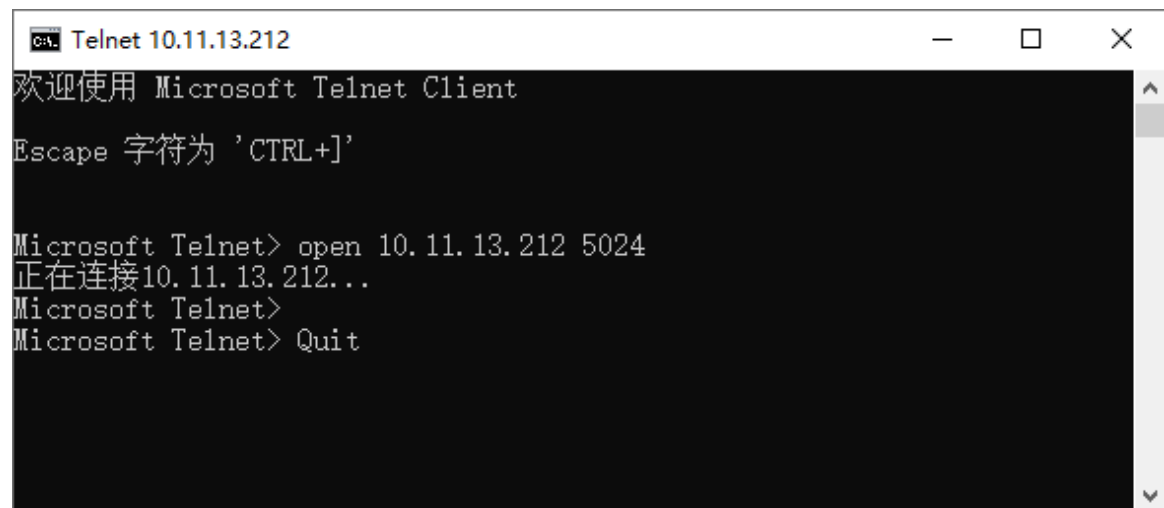
```
CA. Telnet 10.11.13.212
欢迎使用 Microsoft Telnet Client
Escape 字符为 'CTRL+]'
Microsoft Telnet>
```

- To get SCPI prompt again, type open <ip Address> 5024 and press **Enter**:



```
CA. Telnet 10.11.13.212
欢迎使用 Microsoft Telnet Client
Escape 字符为 'CTRL+]'
Microsoft Telnet> open 10.11.13.212 5024
正在连接10.11.13.212...
```

- To close the normal telnet window, type **Quit** and press **Enter**.



```
Telnet 10.11.13.212
欢迎使用 Microsoft Telnet Client
Escape 字符为 'CTRL+]'

Microsoft Telnet> open 10.11.13.212 5024
正在连接10.11.13.212...
Microsoft Telnet>
Microsoft Telnet> Quit
```