RIGOL

DP2000 Series

Programmable Linear DC Power Supply

Programming Guide

Jan. 2023

Guaranty and Declaration

Copyright

Trademark Information

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

Notices

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

Product Certification

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

Contact Us

If you have any problem or requirement when using our products or this manual, please contact RIGOL.

E-mail: service@rigol.com

Website: *http://www.rigol.com*

Section	Description	Page

List of Tables

1 Document Overview

This manual is your guide to control DP2000 series power supply by sending SCPI commands via remote interface. DP2000 series can communicate with the PC via the USB, LAN, or RS232 interface.

TIP

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

Publication Number

PGH09102-1110

Software Version

00.00.01

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

Format Conventions in this Manual

1. **Key**

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

   indicates the "Utility" shortcut key.

2. **Menu**

   The menu function key is denoted by the format of "Menu Name (Bold) + Character Shading" in the manual. For example, **System** indicates the "System" menu option in the operation interface. Tap **System** to access the "System" function menu.

3. **Operation Procedures**

   The ">" denotes the next step of the operation. For example, 
>**Store** indicates first tapping 
, and then tapping **Store**.

Content Conventions in this Manual

The following DP2000 series power supply model is available. Unless otherwise specified, this manual takes DP2031 as an example to illustrate the functions and operation methods of DP2000 series power supply.

Model	Number of Channels	Output Voltage/Current
DP2031	3	Range 1: 32 V/3 A, 32 V/3 A, 6 V/5 A Range 2[1]: 32 V/2 A, 32 V/2 A, 6 V/10 A (optional)

Note[1]: The CH3 of DP2000 series has two ranges: 6 V/5 A and 6 V/10 A (optional). When it switches to 6 V/10 A, both CH1 and CH2 switch to 32 V/2 A.

2 SCPI Introduction

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

Syntax

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

For example,

`:ANALyzer:TYPE` <*type*>

`:ANALyzer:TYPE?`

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

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

`:TRIGger:IN:TYPE` <*d*>,<*type*>

Symbol Description

The following symbols are not sent with the commands.

1. **Braces { }**

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

2. **Vertical Bar |**

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

3. **Square Brackets [ ]**

The contents in the square brackets can be omitted.

4. Angle Brackets < >

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

Parameter Type

1. Bool

The parameter can be set to ON, OFF, 1, or 0. For example,

:SYSTem:BEEPer <*bool*>

:SYSTem:BEEPer?

Wherein, <*bool*> can be set to {{1|ON}|{0|OFF}}. The query returns 1 or 0.

2. Discrete

The parameter can be any of the values listed. For example,

:ANALyzer:TYPE <*type*>

:ANALyzer:TYPE?

Wherein,

- <*type*> can be set to COM|CURR.
- The query returns COM or CURR.

3. Integer

Unless otherwise specified, the parameter can be any integer (NR1 format) within the effective value range.

CAUTION

Do not set the parameter to a decimal, otherwise, errors will occur.

For example,

:TIMEr:GROUP:INDEx <*val*>

:TIMEr:GROUP:INDEx?

Wherein, <*val*> can be set to an integer ranging from 1 to 512. The query returns an integer ranging from 1 to 512.

4. Real

The parameter can be any real number within the effective value range, and this command accepts parameter input in decimal (NR2 format) and scientific notation (NR3 format). For example,

:TIMEr:TEMPlet:INTErval <*time*>

:TIMEr:TEMPlet:INTErval?

Wherein, <*time*> can be set to any real number ranging from 0.01 (0.01 s) to 3600 (3600 s). The query returns a real number in floating point format.

5. ASCII String

The parameter can be the combinations of ASCII characters. For example,

:MEMory:DELete <*filename*>

Wherein, <*filename*> can be set to NEW.RSF.

Command Abbreviation

All the commands are case-insensitive. They can all be in upper case or in lower case. If abbreviation is used, you must input all the capital letters in the command. For example,

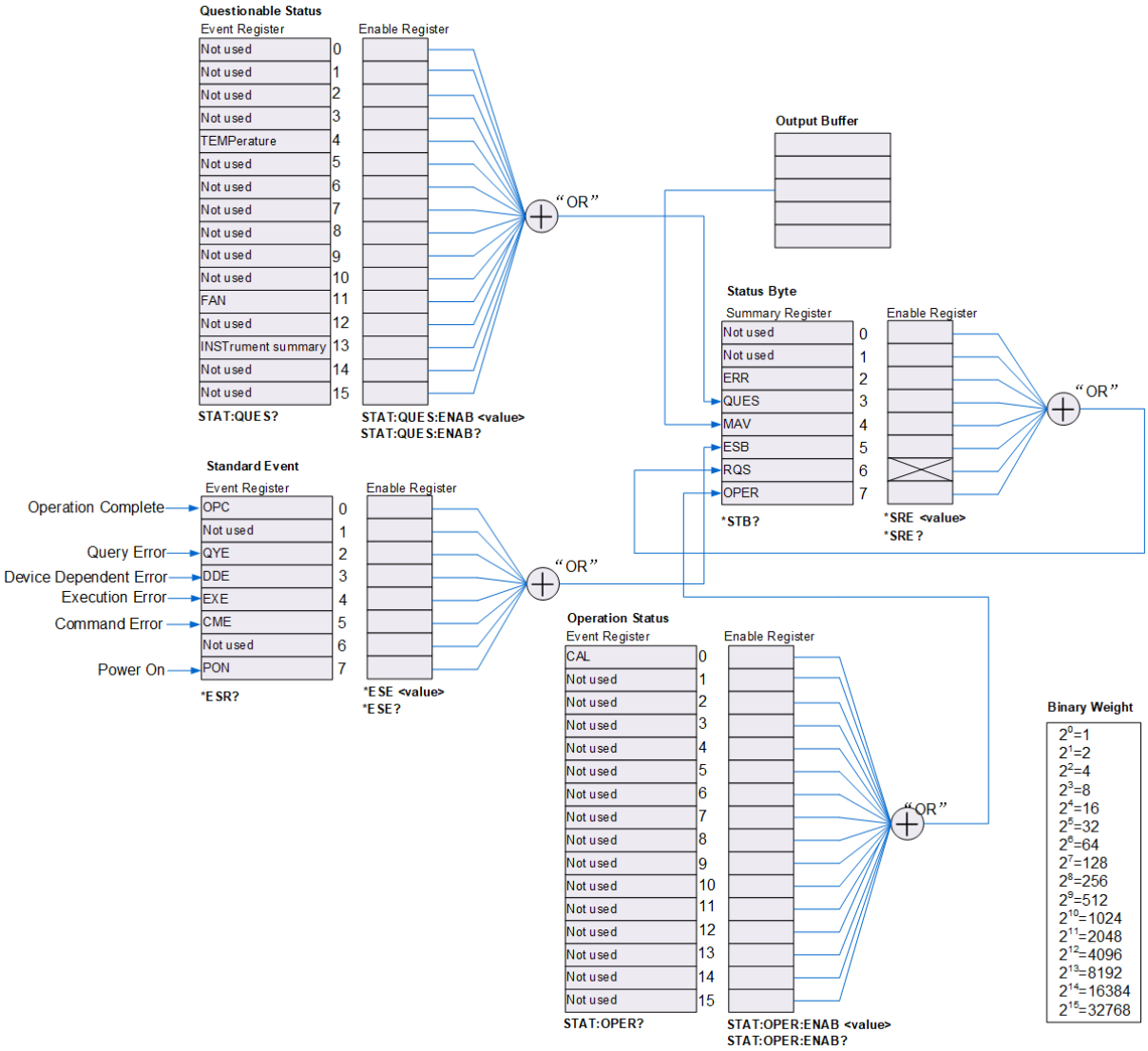
:STSTem:BRIGhtness?

can be abbreviated as

:STST:BRIG?

3 SCPI Status Register

All SCPI instruments implement status registers in the same way. The status system records various instrument conditions in three register groups: the Status Byte register, the Standard Event register, and the Questionable Status register groups. The Status Byte register records high-level summary information reported in other register groups. The figure below illustrates the SCPI status system.



Event Register

An event register is a read-only register that reports defined status within the power supply. Bits in an event register are latched. Once an event bit is set, subsequent state (event state represented by this bit) changes are ignored. Bits in an event register are automatically cleared by a query of the event register (such as **ESR?* or *:STATus:QUEStionable[:EVENt]?*) or by sending the clear status command (**CLS*). The reset command **RST* will not clear bits in event registers. Querying an event

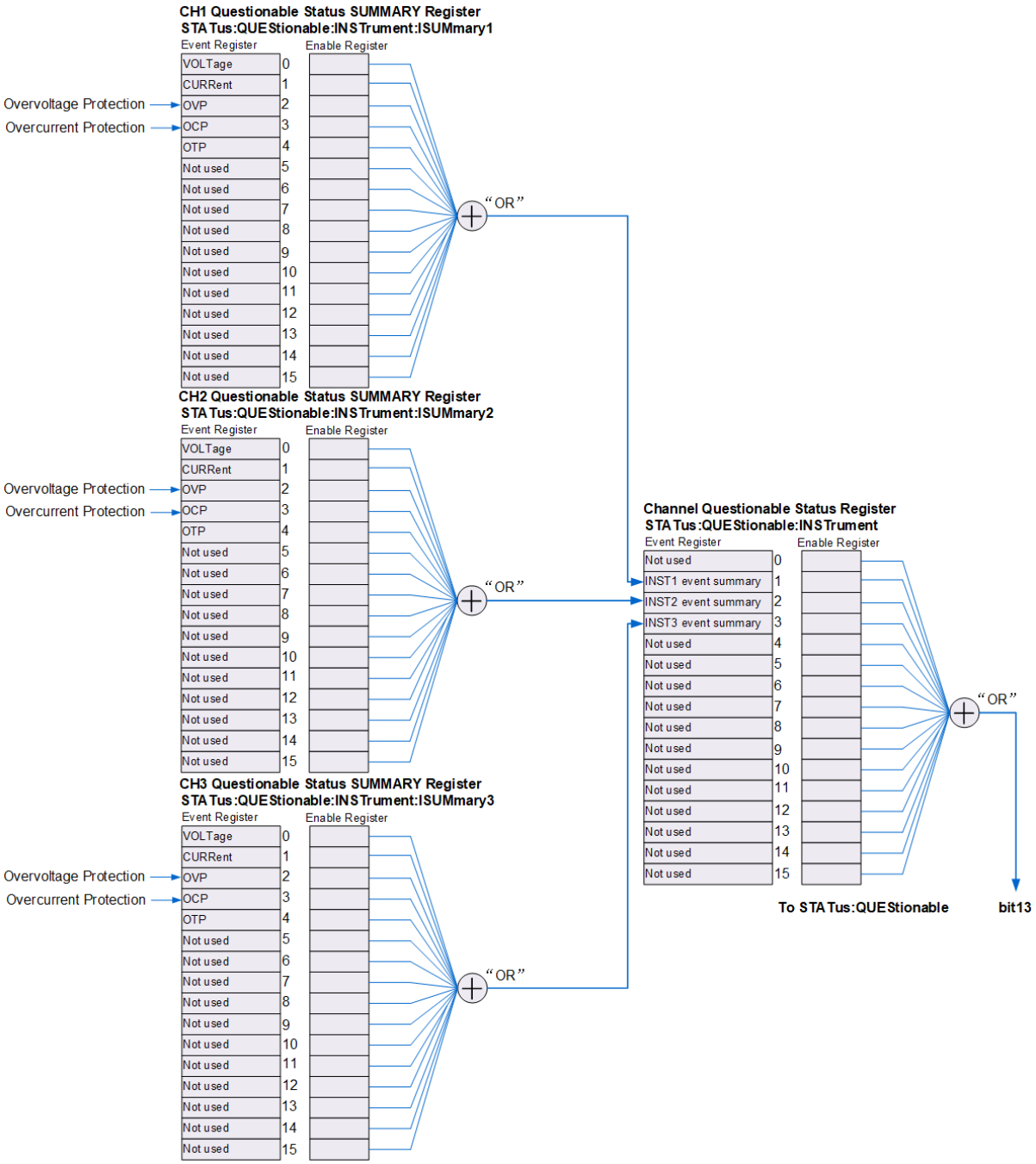
register returns a decimal value of the binary-weighted sum of all bits set in the register.

Enable Register

An enable register is both readable and writable, used to define which status information will be reported to the next level. Querying an enable register will not clear bits in it. Sending the clear status command **CLS* will not clear bits in the enable register (but the command **CLS* does clear the bits in the event registers). To enable bits in an enable register, you must write a decimal value which corresponds to the binary-weighted sum of the bits you wish to enable in the register.

Multiple Logical Output

It is only applicable to multi-channel models. Take DP2031 as an example. The three logical outputs of the power supply include a channel questionable status register and three independent channel questionable status SUMMARY registers (corresponding to the logical outputs of the three channels respectively). The channel questionable status SUMMARY registers report the status of each channel to the channel questionable status register, which in turn reports the channel status to bit13 (ISUM bit) of the Questionable status register.



3.1 Questionable Status Register

The channel questionable status register indicates the channel in which a questionable event occurs. Yet for each specific logical output, the channel questionable status SUMMARY register is a pseudo-questionable status register.

The questionable status register provides information about questionable status of the power supply. bit4 (TEMPerature) reports the over-temperature state; bit11 (FAN) reports the fan failure state and bit13 (INSTrument summary) summaries the questionable output state of any of the three output channels.

Sending *:STATus:QUEStionable[:EVENt]?* will read the register. To use bit13, you must

enable the registers that you wish to summarize with bit13.
Send *:STATus:QUEStionable:INSTrument:ENABle* to enable the channel questionable status register. Then send *:STATus:QUEStionable:INSTrument:ISUMmary[<n>]:ENABle* to enable the corresponding channel questionable status SUMMARY register. The definitions of the bits in the questionable status register of multi-channel models and the decimal values corresponding to their binary weights are as shown in the table below.

Table 3.1 Definitions of the bits in the questionable status register and the decimal values corresponding to their binary weights

Bit		Decimal Value	Definition
0-3	Not used	0	Always be 0.
4	TEMPerature	16	Over-temperature
5-10	Not used	0	Always be 0.
11	FAN	2048	Fan failure.
12	Not used	0	Always be 0.
13	INSTrument summary	8192	Summary information of the channel questionable status register and channel questionable status SUMMARY register group.
14-15	Not used	0	Always be 0.

Channel Questionable Status Register

The channel questionable status register provides information about questionable status of any of the three output channels. bit1 (INST1 event summary), bit2 (INST2 event summary), and bit3 (INST3 event summary) report the information about the questionable states of CH1, CH2, and CH3 respectively.
Sending *:STATus:QUEStionable:INSTrument[:EVENt]?* will read the register. To use the channel questionable status register, you must enable the channel questionable status SUMMARY register.

You can send *:STATus:QUEStionable:INSTrument:ISUMmary[<n>]:ENABle* to enable channel questionable status SUMMARY register. The definitions of the bits in the channel questionable status SUMMARY register and the decimal values corresponding to their binary weights are as shown in the table below.

Table 3.2 Definitions of the bits in the channel questionable status register and the decimal values corresponding to their binary weights

Bit		Decimal Value	Definition
0	Not used	0	Always be 0.
1	INST1 event summary	2	Summary information of CH1 events.
2	INST2 event summary	4	Summary information of CH2 events.
3	INST3 event summary	8	Summary information of CH3 events.
4-15	Not used	0	Always be 0.

Channel Questionable Status SUMMARY Register

There are three channel questionable status Summary registers, one for each channel. These registers provide information about voltage and current regulation as well as overvoltage and overcurrent. bit0 (VOLTage) is set when the voltage becomes unregulated, and bit1 (CURRent) is set when the current becomes unregulated. Sending *:STATus:QUEStionable:INSTrument:ISUMmary[<n>][:EVENt]?* will read the channel questionable status SUMMARY register of the corresponding channel. The definitions of the bits in the channel questionable status SUMMARY register and the decimal values corresponding to their binary weights are as shown in the table below.

Table 3.3 Definitions of the bits in the channel questionable status SUMMARY register and the decimal values corresponding to their binary weights

Bit		Decimal Value	Definition
0	VOLTage	1	The power supply is operating in constant current mode and the voltage is unregulated.
1	CURRent	2	The power supply is operating in constant voltage mode and the current is unregulated.
2	OVP	4	Overvoltage protection occurs.
3	OCP	8	Overcurrent protection occurs.
4	OTP	16	Over-temperature protection occurs.

Bit		Decimal Value	Definition
4-15	Not used	0	Always be 0.

Sending *:STATus:QUEStionable:INSTrument:ISUMmary[<n>]:CONDition?* will query the current operation mode (CC or CV) of the corresponding channel. bit0 true indicates constant current mode; bit1 true indicates constant voltage mode; both bits true indicates that neither the voltage nor the current is regulated, and both bits false indicates that the outputs of the power supply are off.

3.2 Standard Event Register

The standard event register reports the following types of instrument events: power-on detection, command syntax errors, command execution errors, self-test or calibration errors, query errors, or an operation is complete. Any or all of these events can be reported to the bit5 (ESB, Event Summary Bit) of the status byte register through the enable register. To set the enable register mask, you need to write a decimal value to the register using **ESE*. The definitions of the bits in the standard event register and the decimal values corresponding to their binary weights are as shown in the table below.

An error condition (bit2, bit3, bit4, or bit5 of the standard event register) will record one or more errors in the power supply's error queue. You can send *:SYSTem:ERRor[:NEXT]?* to read the error queue.

Table 3.4 Definitions of the bits in the standard event register and the decimal values corresponding to their binary weights

Bit		Decimal Value	Definition
0	OPC	1	Operation complete. All commands prior to and including the **OPC* command have been executed.
1	Not used	0	Always be 0.
2	QYE	4	Query error. The power supply tried to read the output buffer but it was empty, or a new command line was received before a previous query had been read, or both the input and output buffers are full.
3	DDE	8	Device error. A self-test or calibration error occurred.
4	EXE	16	Execution error, including trigger ignore, initialization ignore, setting conflict, data overrange, data too long, and invalid parameter value.
5	CME	32	Command error. A command syntax error occurred.

Bit		Decimal Value	Definition
6	Not used	0	Always be 0.
7	PON	128	Power-on detection. Power has been turned off and on since the last time the event register was read or cleared.

3.3 Status Byte Register

The status byte register reports the status information of other status registers. Query data waiting in the output buffer of the power supply is immediately reported through the bit4 (MAV, Message Available Bit) of the status byte register. Bits in the SUMMARY register of the status byte register are not latched. Clearing an event register will clear the corresponding bits in the status byte SUMMARY register. Reading all messages in the output buffer, including pending queries, will clear the bit4 (MAV, Message Available Bit). The definitions of the bits in the status byte register and the decimal values corresponding to their binary weights are as shown in the table below.

Table 3.5 Definitions of the bits in the status byte register and the decimal values corresponding to their binary weights

Bit		Decimal Value	Definition
0-1	Not used	0	Always be 0.
2	ERR	4	One or more errors have been stored in the Error Queue.
3	QUES	8	One or more bits are set in the questionable status register (bits must be enabled in the enable register).
4	MAV	16	Data is available in the power supply output buffer.
5	ESB	32	One or more bits are set in the standard event register (bits must be enabled in the enable register).
6	RQS	64	The power supply is requesting service.
7	OPER	128	One or more bits are set in the operation event register (bits must be enabled in the enable register).

4 Command System

This chapter introduces the syntax, function, parameter, and usage of each command in A-Z order.

NOTE

- Unless otherwise specified, this manual takes DP2031 as an example to illustrate the commands.
- For the parameter setting command (time, voltage, current, etc.), the instrument can only recognize the numbers, unable to recognize the unit sent together with them. The unit of the parameter is a default one. The table below lists the default units of different parameters.

Type	Default Unit
Time	s
Voltage	V
Current	A
Power	W

4.1 :ANALyzer Commands

:ANALyzer commands are used to set the analyzer parameters, execute analysis, and query the analysis results.

4.1.1 :ANALyzer:COMMon:MEASure:TYPE

Syntax

:ANALyzer:COMMon:MEASure:TYPE <*ch*>[,<*ch*>[,<*ch*>]]

:ANALyzer:COMMon:MEASure:TYPE?

Description

Sets or queries the analysis object of the common analysis function.

Parameter

Name	Type	Range	Default
<ch>	Discrete	{CH1_V|CH1_C|CH1_P|CH2_V|CH2_C|CH2_P|CH3_V|CH3_C|CH3_P}	-

Remarks

You can select one to three items from the "Current", "Voltage", and "Power" items of CH1 to CH3.

Return Format

The query returns one to three items from CH1_V, CH1_C, CH1_P, CH2_V, CH2_C, CH2_P, CH3_V, CH3_C, and CH3_P.

Examples

```
:ANALyzer:COMMon:MEASure:TYPE CH1_V,CH2_P /*Sets the common
analysis objects to the voltage of CH1 and power of CH2.*/
:ANALyzer:COMMon:MEASure:TYPE? /*Queries the common analysis
object. The query returns CH1_V,CH2_P.*/
```

4.1.2 :ANALyzer:CURRent:MEASure:TYPE

Syntax

:ANALyzer:CURRent:MEASure:TYPE <*ch*>[,<*ch*>]

:ANALyzer:CURRent:MEASure:TYPE?

Description

Sets or queries the analysis object of the pulse current analysis.

Parameter

Name	Type	Range	Default
<ch>	Discrete	{CH1|CH2}	-

Remarks

None.

Return Format

The query returns CH1, CH2, or CH1,CH2.

Examples

```
:ANALyzer:CURRent:MEASure:TYPE CH1,CH2 /*Sets the analysis object
of pulse current analysis to CH1 and CH2.*/
:ANALyzer:CURRent:MEASure:TYPE? /*Queries the analysis object of
pulse current analysis. The query returns CH1,CH2.*/
```

4.1.3 :ANALyzer:CURRent:THRE

Syntax

:ANALyzer:CURRent:THRE <*ch*>,<*type*>,<*bool*>,<*val*>

:ANALyzer:CURRent:THRE? <*ch*>,<*type*>

Description

Sets or queries the limit value of positive/negative pulse for the pulse current analysis function.

Parameter

Name	Type	Range	Default
<ch>	Discrete	{CH1|CH2}	-
<type>	Discrete	{UP|LOW}	-
<bool>	Bool	{0|1|ON|OFF}	OFF
<val>	Real	0 to 3.15 A	Positive: 1 A Negative: 0.1 A

Remarks

- **UP** sets <val> for positive pulse. In the analysis process, the analyzer records the number of pulses above the threshold and the most recent pulse width beyond this upper threshold.
- **LOW** sets <val> for negative pulse. In the analysis process, the analyzer records the number of pulses below the threshold and the most recent pulse width below this threshold.

Return Format

The query returns the on/off state of the upper or lower limit and the specific limit value for the selected channel. For example, the query may return 1,1.0000.

Examples

```
:ANALyzer:CURRent:THRE CH1,UP,ON,1 /*Sets the positive pulse
threshold to 1 A for CH1 pulse current analysis.*/
:ANALyzer:CURRent:THRE? CH1,UP /*Queries the positive pulse
threshold for CH1 pulse current analysis. The query returns
1,1.0000.*/
```

4.1.4 :ANALyzer:SAVE:ROUTe

Syntax

:ANALyzer:SAVE:ROUTe <*dest*>

:ANALyzer:SAVE:ROUTe?

Description

Sets or queries the path where the log file is saved.

Parameter

Name	Type	Range	Default
<dest>	ASCII string	Valid storage path	-

Remarks

<dest> sets the specified path in internal/external memory in format of <route>.ROF (e.g. C:/RA.ROF); wherein, <route> indicates the file path and can contain Chinese characters, English letters as well as numbers, and the file extension ".ROF" is the suffix to the filename, which cannot be omitted.

Return Format

The query returns the path where the log file is currently saved, for example, C:/RA.ROF.

Examples

```
:ANALyzer:SAVE:ROUTe C:/RA.ROF /*Sets the current saved path of the log file to C:/RA.ROF.*/
:ANALyzer:SAVE:ROUTe? /*Queries the path where the log file is currently saved. The query returns C:/RA.ROF.*/
```

4.1.5 :ANALyzer:SAVE:STATe

Syntax

:ANALyzer:SAVE:STATe <*bool*>

:ANALyzer:SAVE:STATe?

Description

Sets or queries whether to save the logged data.

Parameter

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

Remarks

- Turn on the analyzer to store the waveform data collected in real time at the current sample rate when the logger is enabled.
- Record at least 1 point to save the file.
- Turn off the analyzer to end the logging. The logged data is automatically saved in the predefined path.

Return Format

The query returns 0 or 1.

Examples

```
:ANALyzer:SAVE:STATe 1 /*Turns on the logger*/
:ANALyzer:SAVE:STATe? /*Queries the on/off state of the logger. The query returns 1.*/
```

4.1.6 :ANALyzer:STATe

Syntax

:ANALyzer:STATe <*bool*>

:ANALyzer:STATe?

Description

Sets or queries the run/stopped state of the analyzer.

Parameter

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

Remarks

None.

Return Format

The query returns 0 or 1.

Examples

```
:ANALyzer:STATe ON /*Turns on the analyzer.*/
:ANALyzer:STATe? /*Queries the on/off state of the analyzer. The query returns 1.*/
```

4.1.7 :ANALyzer:TYPE

Syntax

:ANALyzer:TYPE <*type*>

:ANALyzer:TYPE?

Description

Sets or queries the type of analysis.

Parameter

Name	Type	Range	Default
<type>	Discrete	{COM|CURR}	COM

Remarks

You can send this command to select common analysis (COM) or pulse current analysis (CURR).

Return Format

The query returns COM or CURR.

Examples

```
:ANALyzer:TYPE CURR /*Sets the analysis type to pulse current
analysis.*/
:ANALyzer:TYPE? /*Queries the analysis type. The query returns
CURR.*/
```

4.2 :APPLy Commands

:APPLy command is the most straightforward method to program the power supply via remote interfaces. For multi-channel models, you can select the specified channel and set the voltage and current in a single command. If the setting values are within the parameter ranges of the corresponding channel or range of the specified model, the output voltage and current will change to the setting values immediately after this command is executed. The range and default value of voltage/current corresponding to each channel of DP2031 are shown in the table below.

Table 4.9 Ranges and default values of voltage/current corresponding to each channel of DP2031 series

Channel (Range)			Voltage/Current Available Range	Voltage/Current Default Value
DP2031	Range1	CH1	0 V~32 V/0 V~3 A	0 V/0.1 A

Channel (Range)			Voltage/Current Available Range	Voltage/Current Default Value
		CH2	0 V~32 V/0 V~3 A	0 V/0.1 A
		CH3	0 V~6 V/0 V~5 A	0 V/0.1 A
	Range2 (optional)	CH1	0 V~32 V/0 V~2 A	0 V/0.1 A
		CH2	0 V~32 V/0 V~2 A	0 V/0.1 A
		CH3	0 V~6 V/0 V~10 A	0 V/0.1 A

4.2.1 :APPLy

Syntax

:APPLy [<*source*>[,<*volt*>|<*app*>[,<*curr*>|<*app*>]]]

:APPLy? [<*source*>[,<*option*>]]

Description

Selects the specified channel as the present channel and sets the voltage/current value for this channel.

Queries the voltage/current value for the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-
<volt>	Real	Please refer to *Table 4.9 Ranges and default values of voltage/current corresponding to each channel of DP2031 series*	
<curr>	Real	Please refer to *Table 4.9 Ranges and default values of voltage/current corresponding to each channel of DP2031 series*	
<option>	Discrete	{CURR|VOLT}	-
<app>	Discrete	{MINimum|MAXimum|DEF}	-

Remarks

- In the query command, <source> determines the channel to be queried. If it is omitted, the command queries the present channel.
- <volt> and <curr> determine the voltage and current of the specified channel respectively. If you specify only one value for the parameter, the power supply

regards it as voltage setting value; If you do not specify any value for the parameter, this command only selects the channel and acts as *:INSTrument[:SELEct]*.

- You can substitute "MINimum", "MAXimum", or "DEF" with a specific value for the voltage/current minimum, maximum, or default value. For the voltage/current ranges of each channel (range) of different models, please refer to *Table 4.9 Ranges and default values of voltage/current corresponding to each channel of DP2031 series* .
- <option> determines the object to be queried, voltage or current of the specified channel. If it is omitted, this command queries both the voltage and current values for the specified channel.

Return Format

The query returns a string.

- If only <source> is specified, the query returns the specified channel name, rated voltage/current, the voltage setting value, and current setting value. For example, the query might return CH1:32V/3A,5.000,1.0000.
- When all parameters are omitted, the query returns the voltage setting value and current setting value of the selected channel, for example, 5.000,1.0000.

Examples

```
:APPL CH1,5,1 /*Sets the voltage to 5 V and current to 1 A for
CH1.*/
:APPL? CH1 /*Queries the voltage and current of CH1. The query
returns CH1:32V/3A,5.000,1.0000.*/
```

4.3 IEEE488.2 Commands

4.3.1 *CLS

Syntax

***CLS**

Description

Clears all event registers.

Parameter

None.

Remarks

- You can also send command that queries the event register (*:STATus:QUEStionable[:EVENt]?* or **ESR?*) to clear the corresponding event register.

- The reset command (*RST) or device clear command cannot clear event registers.

Return Format

None.

4.3.2 *ESR?

Syntax

***ESR?**

Description

Queries the event register of the Standard Event register and clears all bits in the register.

Parameter

None.

Remarks

- Executes this command and the query returns a decimal value (corresponding to the binary-weighted sum of all bits set in the register) and clear the status of the register. For definitions of the bits in the Standard Event register and the decimal values corresponding to their binary weights, please refer to *Definitions of the bits in the standard event register and the decimal values corresponding to their binary weights*.

  For example, if query error and execution error currently occur in the instrument, the bit2 (query error bit) and bit4 (execution error bit) in the event register of the Standard Event register are set and this command returns 20 (according to $2^2+2^4=20$).
- The bits in the event register of the Standard Event register are latched and reading the register will clear it. You can also send *CLS to clear the register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits set in the event register of the Standard Event register. For example, the query may return 20.

Examples

```
*ESR? /*Queries the event register of the Standard Event register
and clears all bits in the register. The query returns 20.*/
```

4.3.3 *ESE

Syntax

***ESE** <*enable_value*>

***ESE?**

Description

Enables or queries the bits in the enable register of the Standard Event register.

Parameter

Name	Type	Range	Default
<enable_value>	Character	Refer to *Remarks*	-

Remarks

- The <enable value> is a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Standard Event register. For definitions of the bits in the Standard Event register and the decimal values corresponding to their binary weights, please refer to *Definitions of the bits in the standard event register and the decimal values corresponding to their binary weights*.

  For example, to enable bit2 (query error) and bit4 (execution error) in the enable register of the Standard Event register, set <enable value> to 20 (according to $2^2+2^4=20$).
- Enable the bits in the enable register of the Standard Event register and the system will report the state of the corresponding bit to the Status Byte register.
- When <enable value> is set to 0, executing this command will clear the enable register of the Standard Event register.
- You can also send **PSC* (*PSC 1) to clear the enable register of the Standard Event register at the next power-on.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Standard Event register. For example, the query might return 20.

Examples

```
*ESE 20 /*Enables bit2 (query error) and bit4 (execution error) in
the enable register of the Standard Event register.*/
*ESE? /*Queries the enabled bits in the enable register of the
Standard Event register. The query returns 20.*/
```

4.3.4 *IDN?

Syntax

***IDN?**

Description

Queries the instrument’ s identification string.

Parameter

None.

Remarks

None.

Return Format

The query returns the ID string in the format of Rigol Technologies,<model>,<serial number>,<software version>.

- <model>: the model number.
- <serial number>: the serial number.
- <software version>: the software version.

4.3.5 *OPC

Syntax

***OPC**

***OPC?**

Description

Sets the bit0 (OPC, “Operation Complete” bit) of the Standard Event register after the command is executed.

Queries whether all the previous commands are executed.

Parameter

None.

Remarks

- "Operation Complete" means that all commands prior to and including an *OPC command have been executed.
- Sending the *OPC? command and reading the result can ensure synchronization.
- When you program the desired instrument configuration (by executing the command string), using this command as the last command can determine when the command sequence is completed (when the command sequence is completed, the bit0 (OPC, "Operation Complete" bit) in the event register of the Standard Event register will be set).
- If you send *OPC after a command that loads a query response in the instrument's output buffer (query data), you can use the "OPC" bit to determine when the message is available.

Return Format

The query returns +1 if all the previous commands have been executed.

Examples

```
*OPC /*Sets the bit0 (OPC, "Operation Complete" bit) of the
Standard Event register after the command is executed.*/
*OPC? /*Queries whether the current operation is complete. The
query returns +1.*/
```

4.3.6 *OPT?

Syntax

***OPT?**

Description

Queries the installation status of the options.

Parameter

None.

Remarks

- The options available for DP2000 include CH3 10 A high range mode and 7.5 kSa/s high-speed sampling option.

- To use the optional functions, please order the corresponding options and install them correctly (*:LIC:SET*).

Return Format

The query returns the installation status of the options and different options are separated by ",". The query returns the option name if the option is installed; otherwise, the query has no returned value.

Type	Returned Value
7.5 kSa/s high-speed sampling option	DP2000-HADC
CH3 10 A high range mode	DP2000-10A

For example, the query might return DP2000-HADC,DP2000-10A, indicating that the 2 options mentioned above have been installed.

Examples

```
*OPT? /*Queries the installation status of the options. The query
returns DP2000-HADC,DP2000-10A.*/
```

4.3.7 *PSC

Syntax

***PSC** <*bool*>

***PSC?**

Description

Enables or disables the function of clearing the enable registers of the Status Byte and Standard Event registers at power-on.

Queries the on/off state of the function of clearing the enable registers of the Status Byte and Standard Event registers at power-on.

Parameter

Name	Type	Range	Default
<bool>	Bool	{0|1}	0

Remarks

- *PSC 1 denotes clearing the enable registers of the Status Byte and Standard Event registers at power-on; *PSC 0 denotes that the enable registers of the Status Byte and Standard Event registers will not be affected at power-on.

- You can also send **SRE* (*SRE 0) and **ESE* (*ESE 0) to clear the enable registers of the Status Byte and Standard Event registers respectively.

Return Format

The query returns 0 or 1.

Examples

```
*PSC 1 /*Enables the function of clearing the enable registers of
the Status Byte and Standard Event registers at power-on.*/
*PSC? /*Queries the on/off state of the function of clearing the
registers at power-on. The query returns 1.*/
```

4.3.8 *RCL

Syntax

***RCL** <*n*>

Description

Recalls a previously stored instrument state from the internal memory.

Parameter

Name	Type	Range	Default
<n>	Discrete	{0|1|2|3|4|5|6|7|8|9}	-

Remarks

- This command recalls a previously stored state from the power supply' s internal memory. Using number from 0 to 9 can recall the states named RIGOL0.RSF~RIGOL9.RSF respectively.
- This command is valid only when a state file has been stored in the specified storage location in the internal memory.
- You can also send *:MEMory[:STATe]:LOAD* to recall a previously stored instrument state from the internal memory.

Return Format

None.

Examples

```
*RCL 5 /*Recalls the state file named RIGOL5.RSF stored in the
internal memory.*/
```

4.3.9 *RST

Syntax

***RST**

Description

Restores the power supply to its factory default.

Parameter

None.

Remarks

Executing this command will immediately restore the power supply to its factory default without querying.

Return Format

None.

4.3.10 *SAV

Syntax

***SAV** <*n*>

Description

Saves the current instrument state to the specified location in the internal memory with the specified filename (RIGOLn.RSF).

Parameter

Name	Type	Range	Default
<n>	Discrete	{0|1|2|3|4|5|6|7|8|9}	-

Remarks

- The command saves the current instrument state in the specified location, overwriting the previous state of the same filename (if any). If the state file stored in the specified storage location is locked (*:MEMory:LOCK*), this command is invalid (not overwrite the previous file).
- You can send *:MEMory:STORe* to save the current status to internal/external memory.

Return Format

None.

Examples

```
*SAV 5 /*Saves the current instrument state to the internal memory
with the filename RIGOL5.RSF.*/
```

4.3.11 *SRE

Syntax

***SRE** <*enable_value*>

***SRE?**

Description

Enables bits in the enable register of the Status Byte register.

Queries the enabled bits in the enable register of the Status Byte register.

Parameter

Name	Type	Range	Default
<enable_value>	Discrete	Refer to *Remarks*	-

Remarks

- The <enable value> is a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Status Byte register. For the definitions of the bits in the Status Byte register and their corresponding decimal values, please refer to *Status Byte Register*.

  For example, to enable the bit3 (QUES) and bit4 (MAV) in the Status Byte enable register, set <enable_value> to 24 (2^3+2^4).
- After the bits are enabled, the system sends service request via the bit6 (service request bit) in the Status Byte register.
- When <enable_value> is set to 0, executing this command will clear the enable register of the Status Byte register. You can also send **PSC* (*PSC 1) to clear the enable register of the Status Byte register at the next power-on.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Status Byte register. For example, the query might return +24.

Examples

```
*SRE 24 /*Enables the bit3 (QUES) and bit4 (MAV) in the enable
register of the Status Byte register and enables the service
request.*/
*SRE? /*Queries the enabled bits in the enable register of the
Status Byte register. The query returns +24.*/
```

4.3.12 *STB?

Syntax

***STB?**

Description

Queries the SUMMARY register of the Status Byte register.

Parameter

None.

Remarks

Executes this command and the query returns a decimal value (corresponding to the binary-weighted sum of all bits set in the register). This command does not clear the register. For definitions of the bits in the Status Byte register and the decimal values corresponding to their binary weights, please refer to *Definitions of the bits in the Status Byte register and the decimal values corresponding to their binary weights*. For example, if questionable state currently occurs in the instrument and the service request sent is interrupted, the bit3 (QUES) and bit6 (RQS) in the SUMMARY register of the Status Byte register are set and the query returns 72 (2^3+2^6).

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of all bits set in the SUMMARY register of the Status Byte register. For example, the query might return +72.

Examples

```
*STB? /*Queries the SUMMARY register of the Status Byte register.
The query returns +72.*/
```

4.3.13 *TRG

Syntax

***TRG**

Description

Generates an event trigger.

Parameter

None.

Remarks

- This command is only applicable to the trigger system that has "BUS (software) trigger" as its trigger source.
- When "Bus (software) trigger" is selected, sending this command will trigger the power supply and generate a trigger after the specified delay time.

Return Format

None.

Examples

```
*TRG /*Generates an event trigger.*/
```

4.3.14 *TST?

Syntax

`*TST?`

Description

Queries the self-test result of the instrument.

Parameter

None.

Remarks

The power supply performs a power-on self-test. This commands queries the self-test result.

Return Format

Queries the result of the self-test that the instrument performed. The query returns +0 if it passes and +1 if it fails.

Examples

```
None.
```

4.3.15 *WAI

Syntax

`*WAI`

Description

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

Parameter

None.

Remarks

When "BUS" (Bus trigger, namely software trigger) is selected, this command can ensure synchronization. After the command is executed, the instrument will wait for all the pending operations to complete before executing any other commands.

Return Format

None.

Examples

```
*WAI /*Waits for all the pending operations to complete before
executing any other commands.*/
```

4.4 :INSTrument Commands

:INSTrument commands are used to select the channel to be programmed or query the channel currently selected.

4.4.1 :INSTrument:NSELect

Syntax

:INSTrument:NSELect <*n*>

:INSTrument:NSELect?

Description

Selects the channel to be programmed or queries the channel currently selected.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3}	1

Remarks

- The parameters 1, 2, and 3 represent CH1, CH2, and CH3 respectively.
- This command uses numbers to substitute the channel identifiers in *:INSTrument[:SELEct]*. It functions the same as *:INSTrument[:SELEct]* and *:INSTrument[:SELect]*.

Return Format

The query returns 1, 2, or 3, representing CH1, CH2, and CH3 respectively.

Examples

```
:INST:NSEL 2 /*Selects CH2 as the current channel.*/
:INST:NSEL? /*Queries the channel currently selected. The query
returns 2.*/
```

4.4.2 :INSTrument[:SELect]

Syntax

:INSTrument[:SELect] <*source*>

:INSTrument[:SELect]?

Description

Selects the channel to be programmed or queries the channel currently selected.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	CH1

Remarks

This command functions the same as *:INSTrument:NSELect* and *INSTrument[:SELect]*.

Return Format

The query returns the channel name and its rated voltage/current. For example, the query may return CH1:32V/3A or CH2:32V/3A.

Examples

```
:INST CH2 /*Selects CH2 as the current channel.*/
:INST? /*Queries the channel currently selected. The query returns
CH2:32V/3A.*/
```

4.4.3 :INSTrument[:SELEct]

Syntax

:INSTrument[:SELEct] <*source*>

:INSTrument[:SELEct]?

Description

Selects the channel to be programmed or queries the channel currently selected.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	CH1

Remarks

This command functions the same as *:INSTrument:NSELect* and *:INSTrument[:SELect]*.

Return Format

The query returns the channel name and its rated voltage/current. For example, the query may return CH1:32V/3A or CH2:32V/3A.

Examples

```
:INST CH2 /*Selects CH2 as the current channel.*/
:INST? /*Queries the channel currently selected. The query returns
CH2:32V/3A.*/
```

4.5 :LIC Commands

:LIC commands are used to install options.

4.5.1 :LIC:SET

Syntax

`:LIC:SET` <*license*>

Description

Installs the option.

Parameter

Name	Type	Range	Default
<license>	ASCII string	Refer to *Remarks*	-

Remarks

- Installing the option requires the option license. <License> is a string of fixed characters. For each instrument, the license is unique.
- To acquire the license, you need to purchase the desired option to get the key and then use the key to generate the option license following the steps below.

- Log in to the RIGOL official website (*www.rigol.com*), and click **SERVICE CENTRE** > **SERVICE** > **License Activation** to enter the software license registration interface.
- In the software license registration interface, input the correct key, serial number (tap  > **Help** > **About** to obtain the serial number of the instrument), and verification code. Then click **Generate** to obtain the license file download link. If you need to use the file, please click the link to download the file to the root directory of the USB storage device.

- You can send **OPT?* to query the installation of the specified option.

Return Format

None.

Examples

```
None.
```

4.5.2 :LIC:INSTall

Syntax

`:LIC:INSTall` <*license*>

Description

Installs the option.

Parameter

Name	Type	Range	Default
<license>	ASCII string	Refer to *Remarks*	-

Remarks

- Installing the option requires the option license. <License> is a string of fixed characters. For each instrument, the license is unique.
- To acquire the license, you need to purchase the desired option to get the key and then use the key to generate the option license following the steps below.

- Log in to the RIGOL official website (*www.rigol.com*), and click **SERVICE CENTRE** > **SERVICE** > **License Activation** to enter the software license registration interface.
- In the software license registration interface, input the correct key, serial number (tap > **Help** > **About** to obtain the serial number of the instrument), and verification code. Then click **Generate** to obtain the license file download link. If you need to use the file, please click the link to download the file to the root directory of the USB storage device.

- You can send **OPT?* to query the installation of the specified option.

Return Format

None.

Examples

```
None.
```

4.6 :MEASure Commands

:MEASure commands are used to query the voltage, current, and power measured at the output terminal of the specified channel.

4.6.1 :MEASure[:SCALar]:ALL[:DC]?

Syntax

:MEASure[:SCALar]:ALL[:DC]? [<*source*>]

Description

Queries the voltage, current, and power at the output terminal of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3|SERies|PARallel}	-

Remarks

- **SERies:** Queries the total voltage, current, and power of the present series loop.

- **PARallel:** Queries the total voltage, current, and power of the present parallel loop.

If <source> is omitted, the command queries the voltage, current, and power measured at the output terminal of the channel currently selected.

Yo can also send *:MEASure[:SCALar][:VOLTage][:DC]?*, *:MEASure[:SCALar]:CURRent[:DC]?*, and *:MEASure[:SCALar]:POWEr[:DC]?* to query the voltage, current, and power measured at the output terminal of the specified channel respectively.

Return Format

The query returns the voltage, current, and power (separated by commas) measured at the output terminal of the specified channel. For example, the query might return 2.0000,0.0500,0.100.

Examples

```
:MEAS:ALL? CH1 /*Queries the voltage, current, and power measured
at the output terminal of CH1. The query returns
2.0000,0.0500,0.100.*/
```

4.6.2 :MEASure[:SCALar]:CURRent[:DC]?

Syntax

:MEASure[:SCALar]:CURRent[:DC]? [<*source*>]

Description

Queries the current measured at the output terminal of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3|ALL|SERies|PARallel}	-

Remarks

- **SERies:** Queries the total current of the present series loop.
- **PARallel:** Queries the total current of the present parallel loop.

If <source> is omitted, the command queries the current measured at the output terminal of the specified channel.

You can also send *:MEASure[:SCALar]:ALL[:DC]?* to query the voltage, current, and power measured at the output terminal of the specified channel at the same time.

Return Format

The query returns the the current measured at the output terminal of the specified channel, for example, 0.0500.

Examples

```
:MEAS:CURR? CH1 /*Queries the current measured at the output
terminal of CH1. The query returns 0.0500.*/
```

4.6.3 :MEASure[:SCALar]:CURRent:DATA?

Syntax

:MEASure[:SCALar]:CURRent:DATA? [<*source*>]

Description

Queries the current data measured at the output terminal of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

If <source> is omitted, the command queries the current data measured at the output terminal of the specified channel.

Return Format

The query returns a string starting with **#**.

For example, the query may return #90000046070.047698,0.047938,0.048428,0.048170,...; wherein, #9000004607 is the data block header, and 0.047698,0.047938,0.048428,0.048170,... are the specified current parameters.

- In the format of #NX...X, the data block header is used to describe the length information. For example, in #9000004607, the N is 9, indicating that the 9 numbers following it are intended to describe the data length. That is, the 000004607 indicates the length of this data block (4607 bytes).
- Two adjacent current parameters are separated by ",". For example, 0.047698,0.047938,... can indicate that the first current parameter is 0.047698 A and the second is 0.047938 A.

Examples

```
:MEAS:CURR:DATA? CH1 /*Queries the 512 current parameters measured
at the output terminal of CH1.*/
```

4.6.4 :MEASure[:SCALar]:POWEr[:DC]?

Syntax

:MEASure[:SCALar]:POWEr[:DC]? [<*source*>]

Description

Queries the power measured at the output terminal of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3|ALL|SERies|PARallel}	-

Remarks

- **SERies:** Queries the total power of the present series loop.
- **PARallel:** Queries the total power of the present parallel loop.

If <source> is omitted, the command queries the power measured at the output terminal of the channel currently selected.

You can also send *:MEASure[:SCALar]:ALL[:DC]?* to query the voltage, current, and power measured at the output terminal of the specified channel at the same time.

Return Format

The query returns the power measured on the output terminal of the specified channel, for example, 0.100.

Examples

```
:MEAS:POWE? CH1 /*Queries the power measured on the output terminal
of CH1. The query returns 0.100.*/
```

4.6.5 :MEASure[:SCALar][:VOLTage][:DC]?

Syntax

:MEASure[:SCALar][:VOLTage][:DC]? [<*source*>]

Description

Queries the voltage measured at the output terminal of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3|ALL|SERies|PARallel}	-

Remarks

- **SERies:** Queries the total voltage of the present series loop.
- **PARallel:** Queries the total voltage of the present parallel loop.

If <source> is omitted, the command queries the voltage measured on the output terminal of the specified channel.

You can also send *:MEASure[:SCALar]:ALL[:DC]?* to query the voltage, current, and power measured at the output terminal of the specified channel at the same time.

Return Format

The query returns the the voltage measured at the output terminal of the specified channel, for example, 2.0000.

Examples

```
:MEAS? CH1 /*Queries the voltage measured at the output terminal of
CH1. The query returns 2.0000.*/
```

4.6.6 :MEASure[:SCALar][:VOLTage]:DATA?

Syntax

:MEASure[:SCALar][:VOLTage]:DATA? [<*source*>]

Description

Queries the voltage data measured at the output terminal of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

If <source> is omitted, the command queries the voltage data measured at the output terminal of the specified channel.

Return Format

The query returns a string starting with **#**.

For example, the query may return #90000035830.5763,0.5764,0.5767,0.5765,0.5765,...; wherein, #9000003583 is the data block header, and 0.5763,0.5764,0.5767,0.5765,0.5765,... are the specified voltage parameters.

- In the format of #NX...X, the data block header is used to describe the length information. For example, in #9000003583, the N is 9, indicating that the 9 numbers following it are intended to describe the data length. That is, the 000003583 indicates the length of this data block (3583 bytes).
- Two adjacent voltage parameters are separated by ",". For example, 0.5763,0.5764,... can indicate that the first voltage parameter is 0.5763 V and the second is 0.5764 V.

Examples

```
:MEAS:DATA? CH1 /*Queries the 512 voltage parameters measured at
the output terminal of CH1.*/
```

4.7 :MEMory Commands

:MEMory commands are used to save the file to the specified location in internal/external memory, and delete, read, lock or unlock the file stored in the specified storage location in internal memory. This series power supply allows you to save the following five type of files in internal memory.

1. **State File (RSF)** stores the current system state, including the voltage, current, OVP, OCP and track function state of each channel as well as the system parameters.
2. **Arb File (RTF)** stores the Arb parameters edited (the voltage, current, and time of each group of parameters).
3. **Bitmap File (BMP)** stores the screen capture image file.
4. **Log File (ROF)** stores the waveform data collected in real time at the current sample rate when the Analyzer is running and the Logger is turned on.
5. **Calibration File (CLF)** stores calibration parameters.

4.7.1 :MEMory:CATalog?

Syntax

`:MEMory:CATalog?`

Description

Queries all the files and folders in the current path.

Parameter

None.

Remarks

C disk cannot store folders.

Return Format

The query returns the names of all the files and folders (separated by commas). For example, the query may return RIGOL0.BMP,cc.RSF; wherein, RIGOL0.BMP represents bitmap file, and cc.RSF represents state file.

Examples

```
:MEMory:CATalog? /*Queries the names of files in the current path.
The query returns RIGOL0.BMP,cc.RSF.*/
```

4.7.2 :MEMory:CDIRectory

Syntax

:MEMory:CDIRectory <*directory_name*>

:MEMory:CDIRectory?

Description

Sets or queries the current directory.

Parameter

Name	Type	Range	Default
<directory_name>	ASCII string	Refer to *Remarks*	-

Remarks

- The parameter <directory_name> must be valid.
- Valid external directories include the external memory (D:/ and E:/) and the folders (such as D:/RIGOL) in the external memory.

Return Format

The query returns the current directory, for example, C:/.

Examples

```
:MEMory:CDIRectory C:/ /*Sets the current directory to C disk.*/
:MEMory:CDIRectory? /*Queries the current directory. The query
returns C:/.*/
```

4.7.3 :MEMory:DELete

Syntax

:MEMory:DELete <*filename*>

Description

Deletes the specified file and empty folders in the current directory.

Parameter

Name	Type	Range	Default
<filename>	ASCII string	Refer to *Remarks*	-

Remarks

- <filename> is the name of the file to be deleted (the filename uses the file type as the suffix, for example, STA.RSF) or the name of the empty folder.
- This command is valid only when the current directory contains the specified file or an empty folder.
- The folder cannot be deleted if it is not empty and a prompt message "Unknown error" will be displayed. The command is invalid if the file is locked (*:MEMory:LOCK*).

Return Format

None.

Examples

```
:MEM:DEL NEW.RSF /*Deletes the state file named NEW in the current directory.*/
```

4.7.4 :MEMory:DISK?

Syntax

:MEMory:DISK?

Description

Queries the available external storage disk(s).

Parameter

None.

Remarks

None.

Return Format

The query returns the available disk(s), for example, D:/, E:/. If there is no available external disk, the query returns NONE.

Examples

```
:MEM:DISK? /*Queries the available external memory disk(s). The
query returns D:/, E:/.*/
```

4.7.5 :MEMory:LOAD

Syntax

:MEMory:LOAD <*filename*>

Description

Reads the specified file stored in the current directory, including state file (.RSF) and Arb file (.RTF).

Parameter

Name	Type	Range	Default
<filename>	ASCII string	Filenames (with suffix) of the files stored in the current directory	-

Remarks

- This command is valid only when the file has been stored in the specified storage location.
- You can also use **RCL* to read the specified state file stored in internal memory.

Return Format

None.

Examples

```
:MEM:LOAD NEW.RSF /*Reads the file named NEW.RSF in the current
directory.*/
```

4.7.6 :MEMory:LOCK

Syntax

:MEMory:LOCK <*filename*>,<*bool*>

:MEMory:LOCK? <*filename*>

Description

Locks or unlocks the specified file stored in C disk, or queries whether the specified file in C disk is locked.

Parameter

Name	Type	Range	Default
<bool>	Bool	{0|1|ON|OFF}	0
<filename>	ASCII string	Refer to *Remarks*	-

Remarks

- <filename> must be an existing filename with suffix in C disk.
- Only C disk supports file lock function. The command is valid only when the specified file exists.
- You can only read the locked file but are not allowed to copy and delete the locked file.

Return Format

The query returns 0 or 1.

Examples

```
:MEM:LOCK NEW.RSF,ON /*Locks the file named NEW.RSF in C disk.*/
:MEM:LOCK? NEW.RSF /*Queries whether the file named NEW.RSF in C
disk is locked. The query returns 1.*/
```

4.7.7 :MEMory:MDIRectory

Syntax

:MEMory:MDIRectory <*folder_name*>

Description

Creates a new folder in the current directory.

Parameter

Name	Type	Range	Default
<folder_name>	ASCII string	Refer to *Remarks*	-

Remarks

- <folder_name> is the name of the new folder, which can contain 255 characters in maximum. The name can consist of Chinese characters, English letters, and numbers (one Chinese character takes two bytes).
- Folders cannot be created in C disk. If the current path is C disk, send this command and a prompt message will be displayed.

Return Format

None.

Examples

```
:MEMory:MDIRectory NEW /*Creates a folder named NEW in the current directory.*/
```

4.7.8 :MEMory:STORe

Syntax

:MEMory:STORe <*filename*>

Description

Saves the file including instrument state file (.RSF) and Arb file (.RTF) with the specified filename in the current directory.

Parameter

Name	Type	Range	Default
<filename>	ASCII string	Refer to *Remarks*	-

Remarks

- <filename> is the filename with the file extension of .RSF/.RTF. It can contain 125 characters in maximum. The name can consist of Chinese characters, English letters as well as numbers (one Chinese character takes two bytes).
- The command overwrites the previously stored file (if any) in the current directory. If the file stored in the specified location is locked (*:MEMory:LOCK*), this command is invalid (not overwrite the original file directly).

- The storage path of the log file is specified by *:ANALyzer:SAVE:ROUTe*.
- You can also send **SAV* to save the current instrument state to internal memory.

Return Format

None.

Examples

```
:MEM:STOR NEW.RSF /*Saves the state file in the current location
and sets the filename to NEW.RSF.*/
```

4.7.9 :MEMory:VALid?

Syntax

:MEMory:VALid? <*filename*>

Description

Queries whether the specified file is stored in the current directory.

Parameter

Name	Type	Range	Default
<filename>	ASCII string	Refer to *Remarks*	-

Remarks

<filename> is the filename with suffix, which can contain Chinese characters, English letters as well as numbers.

Return Format

The query returns 0 or 1.

Examples

```
:MEM:STOR NEW.RSF /*Saves a state file named NEW in the current
directory.*/
:MEM:VAL? NEW.RSF /*Queries whether the the state file named NEW is
stored in the current directory. The query returns 1.*/
```

4.8 :OUTPut Commands

:OUTPut commands are used to enable or disable channel output, OVP/OCP function, track function, and Sense function, query the channel output mode, as well as set and query the related information of overvoltage/overcurrent protection. The range and default value of overvoltage/overcurrent protection corresponding to each channel of DP2031 are shown in the table below.

Table 4.35 Range and default value of overvoltage/overcurrent protection

Channel			OVP/OCP Available Range	OVP/OCP Default Value
DP2031	Range1	CH1 (32 V/3 A)	1 mV to 35.2 V/1 mA to 3.3 A	35.2 V/3.3 A
		CH2 (32 V/3 A)	1 mV to 35.2 V/1 mA to 3.3 A	35.2 V/3.3 A
		CH3 (6 V/5 A)	1 mV to 6.6 V/1 mA to 5.5 A	6.6 V/5.5 A
	Range2 (Optional)	CH1 (32 V/2 A)	1 mV to 35.2 V/1 mA to 2.2 A	35.2 V/2.2 A
		CH2 (32 V/2 A)	1 mV to 35.2 V/1 mA to 2.2 A	35.2 V/2.2 A
		CH3 (6 V/10 A)	1 mV to 6.6 V/1 mA to 11 A	6.6 V/11 A

4.8.1 :OUTPut:CVCC?

Syntax

`:OUTPut:CVCC?` [<*source*>]

Description

Queries the output mode (CV, CC, or UR) for the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- This series power supply has three output modes: constant voltage (CV), constant current (CC), and unregulated (UR). In CV mode, the output voltage equals to the voltage setting value, and the output current is determined by the load; whereas in CC mode, the output current equals to the current setting value, and the output voltage is determined by the load. UR mode is the unregulated mode between CV and CC mode.
- If <source> is omitted, the command queries the output mode of the current channel.

Return Format

The query returns CV, CC, or UR.

Examples

```
:OUTPut:CVCC? CH1 /*Queries the current output mode of CH1. The
query returns CV.*/
```

4.8.2 :OUTPut:MODE?

Syntax

:OUTPut:MODE? [<*source*>]

Description

Queries the output mode (CV, CC, or UR) for the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- This series power supply has three output modes:constant voltage (CV), constant current (CC), and unregulated (UR). In CV mode, the output voltage equals to the voltage setting value, and the output current is determined by the load; whereas in CC mode, the output current equals to the current setting value, and the output voltage is determined by the load. UR mode is the unregulated mode between CV mode and CC mode.
- If <source> is omitted, the command queries the output mode of the current channel.

Return Format

The query returns CV, CC, or UR.

Examples

```
:OUTP:MODE? CH1 /*Queries the current output mode of CH1. The query
returns CV.*/
```

4.8.3 :OUTPut:OCP:ALAR?

Syntax

:OUTPut:OCP:ALAR? [<*source*>]

Description

Queries whether an overcurrent protection (OCP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- Overcurrent protection (OCP) indicates that the output is disabled automatically when the actual output current reaches the OCP level.
- You can send *:OUTPut:OCP:CLEar* to clear the OCP event that occurred on the specified channel.
- If <source> is omitted, the command queries the output mode of the current channel.

Return Format

The query returns 0 or 1.

Examples

```
:OUTPut:OCP:ALAR? CH1 /*Queries whether an overcurrent protection
(OCP) event occurred on CH1.*/
```

4.8.4 :OUTPut:OCP:QUES?

Syntax

:OUTPut:OCP:QUES? [<*source*>]

Description

Queries whether an overcurrent protection (OCP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- Overcurrent protection (OCP) indicates that the output is disabled automatically when the actual output current reaches the OCP level.

- You can send *:OUTPut:OCP:CLEar* to clear the OCP event that occurred on the specified channel.
- If <source> is omitted, the command queries the output mode of the current channel.

Return Format

The query returns 1 or 0.

Examples

```
:OUTP:OCP:QUES? CH1 /*Queries whether an OCP event occurred on
CH1.*/
```

4.8.5 :OUTPut:OCP:CLEar

Syntax

:OUTPut:OCP:CLEar [<*source*>]

Description

Clears an overcurrent protection (OCP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- Before executing the command, make sure to remove the condition that caused the overcurrent protection on the specified channel (you can either decrease the output current to below the OCP level or increase the OCP level to above the output current).
- If <source> is omitted, the command clears an OCP event occurred on the current channel.
- You can also send *[:SOURce[<n>]]:CURRent:PROTection:CLEar* to clear an OCP event that occurred on the specified channel and enable the channel output.
- You can send *:OUTPut:OCP:ALAR?* or *:OUTPut:OCP:QUES?* to query whether an OCP event has occurred on the specified channel.

Return Format

None.

Examples

```
:OUTP:OCP:QUES? CH1 /*Queries whether an OCP event occurred on CH1.
The query returns YES.*/
:OUTP:OCP:CLE CH1 /*Clears an OCP event that occurred on CH1.*/
```

```
:OUTP:OCP:QUES? CH1 /*Queries whether an OCP event occurred on CH1.
The query returns NO.*/
```

4.8.6 :OUTPut:OCP:DELay

Syntax

:OUTPut:OCP:DELay [<*source*>,]{<*value*>|<*lim*>}

:OUTPut:OCP:DELay? [<*source*>][,<*lim*>]

Description

Sets or queries the time in milliseconds (default) that the overcurrent protection (OCP) is temporarily disabled.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-
<value>	Real	0 to 1000 ms	10 ms
<lim>	Discrete	{MINimum|MAXimum}	-

Remarks

- If <source> is omitted, the command executes the corresponding operation on the current channel.
- You can select "MINimum" to set the minimum OCP delay or "MAXimum" to set the maximum OCP delay.

Return Format

The query returns the OCP delay, for example, 200ms.

Examples

```
:OUTPut:OCP:DELay CH1 200 /*Sets the OCP delay to 200 ms for CH1.*/
:OUTPut:OCP:DELay? CH1 /*Queries the OCP delay for CH1. The query
returns 200ms.*/
```

4.8.7 :OUTPut:OCP[:STATe]

Syntax

:OUTPut:OCP[:STATe] [<*source*>,]<*bool*>

:OUTPut:OCP[:STATe]? [<*source*>]

Description

Sets or queries the on/off state of the overcurrent protection (OCP) function for the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-
<bool>	Bool	{0|1|ON|OFF}	0

Remarks

- When OCP is enabled, the output is disabled automatically if the output current reaches the OCP level (*:OUTPut:OCP:VALue*) currently set. You can send *:OUTPut:OCP:QUES?* or *:OUTPut:OCP:ALAR?* to query whether an OCP event occurred on the specified channel.
- If <source> is omitted, the command executes the corresponding operation on the current channel.

Return Format

The query returns 0 or 1.

Examples

```
:OUTP:OCP CH1, 1 /*Enables the OCP function for CH1.*/
:OUTP:OCP? CH1 /*Queries the on/off state of the OCP function of
CH1. The query returns 1.*/
```

4.8.8 :OUTPut:OCP:VALue

Syntax

:OUTPut:OCP:VALue [<*source*>,]{<*value*>|<*lim*>}

:OUTPut:OCP:VALue? [<*source*>][,<*lim*>]

:OUTPut:OCP:VALue? [<*lim*>]

Description

Sets or queries the overcurrent protection (OCP) level of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Name	Type	Range	Default
<value>	Real	Please refer to *Table 4.35 Range and default value of overvoltage/overcurrent protection*	
<lim>	Discrete	{MINimum|MAXimum}	-

Remarks

- When OCP is enabled, the output is disabled automatically if the actual output current reaches the OCP level currently set. You can send *:OUTPut:OCP:QUES?* or *:OUTPut:OCP:ALAR?* to query whether an OCP event occurred on the specified channel.
- If <source> is omitted, the command executes the corresponding operation on the current channel.
- You can select "MINimum" to set the minimum OCP level or "MAXimum" to set the maximum OCP level.
- You can also send *[:SOURce[<n>]]:CURRent:PROTection[:LEVel]* to set the OCP level for the specified channel.

Return Format

The query returns the OCP level, for example, 5.0000.

Examples

```
:OUTP:OCP:VAL CH1,5 /*Sets the OCP level of CH1 to 5 A.*/
:OUTP:OCP:VAL? CH1 /*Queries the OCP level of CH1. The query
returns 5.0000.*/
```

4.8.9 :OUTPut:OVP:ALAR?

Syntax

:OUTPut:OVP:ALAR? [<*source*>]

Description

Queries whether an overvoltage protection (OVP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- The overvoltage protection (OVP) function disables the output automatically when the actual output voltage reaches the OVP level.

- If <source> is omitted, the command queries whether an OVP event occurred on the current channel.
- You can send *:OUTPut:OVP:CLEar* to clear the OVP event that occurred on the specified channel.

Return Format

The query returns 1 or 0.

Examples

```
:OUTP:OVP:ALAR? CH1 /*Queries whether an OVP event occurred on
CH1.*/
```

4.8.10 :OUTPut:OVP:QUES?

Syntax

:OUTPut:OVP:QUES? [<*source*>]

Description

Queries whether an overvoltage protection (OVP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- The overvoltage protection (OVP) function disables the output automatically when the actual output voltage reaches the OVP level.
- If <source> is omitted, the command queries whether an OVP event occurred on the current channel.
- You can send *:OUTPut:OVP:CLEar* to clear the OVP event that occurred on the specified channel.

Return Format

The query returns 1 or 0.

Examples

```
:OUTP:OVP:QUES? CH1 /*Queries whether an OVP event occurred on
CH1.*/
```

4.8.11 :OUTPut:OVP:CLEar

Syntax

`:OUTPut:OVP:CLEar` [<*source*>]

Description

Clears an overvoltage protection (OVP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-

Remarks

- Before executing the command, make sure to remove the condition that caused the overvoltage protection on the specified channel (you can either decrease the output voltage to below the OVP level or increase the OVP level to be greater than the output voltage).
- If <source> is omitted, the command clears an OVP event that occurred on the current channel.
- You can also send *[:SOURce[<n>]]:VOLTage:PROTection:CLEar* to clear an OVP event that occurred on the specified channel and enable the channel output.
- You can send *:OUTPut:OVP:QUES?* or *:OUTPut:OVP:ALAR?* to query whether an OVP event has occurred on the specified channel.

Return Format

None.

Examples

```
:OUTP:OVP:CLE CH1 /*Clears an OVP event that occurred on CH1.*/
```

4.8.12 :OUTPut:OVP[:STATe]

Syntax

`:OUTPut:OVP[:STATe]` [<*source*>,]<*bool*>

`:OUTPut:OVP[:STATe]?` [<*source*>]

Description

Sets or queries the on/off state of the overvoltage protection (OVP) function of the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-
<bool>	Bool	{0|1|ON|OFF}	0

Remarks

- When OVP is enabled, the output is disabled automatically if the output voltage reaches the OVP level (*:OUTPut:OVP:VALue*) currently set. You can send *:OUTPut:OVP:QUES?* or *:OUTPut:OVP:ALAR?* to query whether an OVP event occurred on the specified channel.
- If <source> is omitted, the command executes the corresponding operation on the current channel.

Return Format

The query returns 1 or 0.

Examples

```
:OUTP:OVP CH1,1 /*Enables the OVP function for CH1.*/
:OUTP:OVP? CH1 /*Queries the OVP on/off state for CH1. The query
returns 1.*/
```

4.8.13 :OUTPut:OVP:VALue

Syntax

:OUTPut:OVP:VALue [<*source*>,]{<*value*>|<*lim*>}

:OUTPut:OVP:VALue? [<*source*>][,<*lim*>]

:OUTPut:OVP:VALue? [<*lim*>]

Description

Sets or queries the overcurrent protection (OVP) level for the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3}	-
<value>	Real	Please refer to *Table 4.35 Range and default value of overvoltage/overcurrent protection*	
<lim>	Discrete	{MINimum|MAXimum}	-

Remarks

- When OVP is enabled, the output is disabled automatically if the output voltage reaches the OVP level currently set. You can send *:OUTPut:OVP:QUES?* or *:OUTPut:OVP:ALAR?* to query whether an OVP event occurred on the specified channel.
- If <source> is omitted, the command sets or queries the OVP level of the current channel.
- You can select "MINimum" to set the minimum OVP level or "MAXimum" to set the maximum OVP level.
- You can also send *[:SOURce[<n>]]:VOLTage:PROTection[:LEVel]* to set the OVP level for the specified channel.

Return Format

The query returns the OVP level, for example, 8.800.

Examples

```
:OUTP:OVP:VAL CH1,8.8 /*Sets the OVP level of CH1 to 8.8 V.*/
:OUTP:OVP:VAL? CH1 /*Queries the OVP level of CH1. The query
returns 8.800.*/
```

4.8.14 :OUTPut:PAIR

Syntax

:OUTPut:PAIR <*type*>

:OUTPut:PAIR?

Description

Sets or queries the channel connection mode.

Parameter

Name	Type	Range	Default
<type>	Discrete	{OFF|PARallel|SERies}	OFF

Remarks

- **OFF:** CH1 and CH2 are independent of each other.
- **PARallel:** CH1 and CH2 are connected in parallel (internal).
- **SERies:** CH1 and CH2 are connected in series (internal).

Return Format

The query returns the connection mode, for example, SERIES.

Examples

```
:OUTPut:PAIR SERies /*Connects CH1 and CH2 in series (internal).*/
:OUTPut:PAIR? /*Queries the connection mode. The query returns
SERIES.*/
```

4.8.15 :OUTPut[:STATe]

Syntax

:OUTPut[:STATe] [<*source*>,]<*bool*>

:OUTPut[:STATe]? [<*source*>]

Description

Sets or queries the on/off state for the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3|ALL}	-
<bool>	Bool	{0|1|ON|OFF}	0

Remarks

- Before enabling the channel output, please make sure that the current setting will not affect the devices connected to the power supply.
- If <source> is omitted, the command executes the corresponding operation on the current channel.

Return Format

The query returns 1 or 0.

Examples

```
:OUTP CH1,ON /*Enables the output for CH1.*/
:OUTP? CH1 /*Queries whether the CH1 output is enabled. The query
returns 1*/
```

4.8.16 :OUTPut:TRACk[:STATe]

Syntax

:OUTPut:TRACk[:STATe] <*bool*>

:OUTPut:TRACk[:STATe]?

Description

Sets or queries the on/off state of tracking function.

Parameter

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

Remarks

- For the two channels (CH1 and CH2) that support this mode, changes made on one channel (voltage/current setting value, OVP/OCP level and on/off status) are applied to the other channel.
- By default, the tracking function is disabled. It is usually used to provide symmetric voltage for the operational amplifier or other circuits.
- The tracking function only tracks the voltage/current setting value. The actual output voltage/current will not be affected.

Return Format

The query returns 1 or 0.

Examples

```
:OUTP:TRAC ON /*Enables the tracking function for CH2.*/
:OUTP:TRAC? /*Queries the on/off state of the tracking function.
The query returns 1.*/
```

4.9 :SOURce Commands

:SOURce commands are used to set the voltage, current, OVP, and OCP of the specified channel. While the :APPLy command offers the most straightforward method to program the power supply via remote interfaces, :SOURce commands give you more flexibility to change individual parameters.

4.9.1 [:SOURce[<n>]]:CURRent[:LEVel][:IMMediate][:AMPLitude]

Syntax

[:SOURce[<*n*>]**]:CURRent[:LEVel][:IMMediate][:AMPLitude]** {<*current*>|<*lim*>|<*amp*>}

[:SOURce[<*n*>]**]:CURRent[:LEVel][:IMMediate][:AMPLitude]?** [<*lim*>]

Description

Sets or queries the current of the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-
<current>	Real	0 to the maximum current value of the specified channel	0.1 A
<lim>	Discrete	{MINimum|MAXimum|DEFault}	-
<amp>	Discrete	{UP|DOWN}	-

Remarks

- If [:SOURce[<n>]] or [<n>] is omitted, the command sets the corresponding parameter of the channel currently selected.
- When <current> is selected, the command directly sets the current value of the specified channel. When MINimum, MAXimum, or DEFault is selected, the command sets the minimum, maximum, or default current value within the available range for the specified channel. When UP or DOWN is selected, the command steps up or down the current according to the step size set in *[:SOURce[<n>]]:CURRent[:LEVel][:IMMediate]:STEP[:INCRement]*.
- You can also send *:APPLy* to set the current for the specified channel.

Return Format

The query returns the current, for example, 1.5000.

Examples

```
:CURR 1.5 /*Sets the current to 1.5 A for the channel currently
selected .*/
:CURR? /*Queries the current of the channel currently selected. The
query returns 1.5000.*/
```

4.9.2 [:SOURce[<n>]]:CURRent[:LEVel][:IMMediate]:STEP[:INCRement]

Syntax

[:SOURce[<*n*>]**]:CURRent[:LEVel][:IMMediate]:STEP[:INCRement]** {<*numeric value*>|<*def*>}

[:SOURce[<*n*>]**]:CURRent[:LEVel][:IMMediate]:STEP[:INCRement]?** [<*def*>]

Description

Sets or queries the step size of current for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-
<numeric value>	Real	0 to the maximum current value of the specified channel	Refer to *Remarks*
<def>	Discrete	{DEFault}	-

Remarks

- If [:SOURce[<n>]] or [<n>] is omitted, the command sets the corresponding parameter of the channel currently selected.
- <numeric value> is the step size specified. DEFault is the default value. The default values of <numeric value> are as shown in the table below.

Channel	Default
CH1	0.0001 A
CH2	0.0001 A
CH3	0.001 A

- Select UP or DOWN and executes *[:SOURce[<n>]:CURRent[:LEVel][:IMMediate][:AMPLitude]*, and the instrument will step up or down the current by the step size set in this command.

Return Format

The query returns the step size, for example, 0.1000.

Examples

```
:CURR:STEP 0.1 /*Sets the step size to 0.1 A for the selected
channel.*/
:CURR:STEP? /*Queries the step size for the selected channel. The
query returns 0.1000.*/
```

4.9.3 [:SOURce[<n>]]:CURRent:PROTection:CLEar

Syntax

[:SOURce[<*n*>]:CURRent:PROTection:CLEar

Description

Clears an OCP event that occurred on the specified channel and enables the output of the corresponding channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-

Remarks

- You can send *[:SOURce[<n>]]:CURRent:PROTection:TRIPped?* to query whether an OCP event occurred on the specified channel.
- Before executing the command, make sure to remove the condition that caused the overcurrent protection on the specified channel (you can either decrease the output current to below the OCP level or increase the OCP level to be greater than the output current). Execute this command to clear an OCP event that occurred on the specified channel and enable the output of the corresponding channel.
- If [:SOURce[<n>]] or [<n>] is omitted, the command clears an OCP event occurred on the current channel.
- You can send *:OUTPut:OCP:CLEar* command to clear an OCP event that occurred on the specified channel.

Return Format

None.

Examples

```
:CURR:PROT:TRIP? /*Queries whether an OCP event occurred on the
current channel. The query returns 1.*/
:CURR:PROT:CLE /*Clears an OCP event that occurred on the current
channel.*/
:CURR:PROT:TRIP? /*Queries whether an OCP event occurred on the
current channel. The query returns 0.*/
```

4.9.4 [:SOURce[<n>]]:CURRent:PROTection[:LEVel]

Syntax

[:SOURce[<*n*>]**]:CURRent:PROTection[:LEVel]** {<*current*>|<*lim*>}

[:SOURce[<*n*>]**]:CURRent:PROTection[:LEVel]?** [<*lim*>]

Description

Sets or queries the overcurrent protection (OCP) level for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-

Name	Type	Range	Default
<current>	Real	Please refer to *Table 4.35 Range and default value of overvoltage/overcurrent protection*	
<lim>	Discrete	{MINimum|MAXimum}	-

Remarks

- When OCP (*[:SOURce[<n>]]:CURRent:PROTection:STATe*) is enabled, the output is disabled automatically if the actual output current reaches the OCP level currently set. You can send *[:SOURce[<n>]]:CURRent:PROTection:TRIPped?* to query whether an OCP event occurred on the specified channel.
- If [:SOURce[<n>]] or [<n>] is omitted, the command sets the corresponding parameter of the channel currently selected.
- You can also send *:OUTPut:OCP:VALue* to set the OCP level for the specified channel.

Return Format

The query returns the OCP level, for example, 2.0000.

Examples

```
:CURR:PROT 2 /*Sets the OCP level to 2 A for the current channel.*/
:CURR:PROT? /*Queries the OCP level for the current channel. The
query returns 2.0000.*/
```

4.9.5 [:SOURce[<n>]]:CURRent:PROTection:STATe

Syntax

[:SOURce[<*n*>]**]:CURRent:PROTection:STATe** <*bool*>

[:SOURce[<*n*>]**]:CURRent:PROTection:STATe?**

Description

Sets or queries the on/off state of the overcurrent protection (OCP) function for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-
<bool>	Bool	{ON|OFF|1|0}	0

Remarks

- When OCP is enabled, the output is disabled automatically if the actual output current reaches the OCP level currently set. You can send *[:SOURce[<n>]]:CURRent:PROTection:TRIPped?* to query whether an OCP event occurred on the specified channel.
- If [:SOURce[<n>]] or [<n>] is omitted, the command sets the corresponding parameter of the channel currently selected.
- You can also send *:OUTPut:OCP[:STATe]* to enable or disable the OCP function for the specified channel.
- You can send *[:SOURce[<n>]]:CURRent:PROTection[:LEVel]* to query the OCP level for the specified channel.

Return Format

The query returns 0 or 1.

Examples

```
:CURR:PROT:STAT ON /*Enables the OCP function for the current
channel.*/
:CURR:PROT:STAT? /*Queries the on/off state of the OCP function for
the current channel. The query returns 1.*/
```

4.9.6 [:SOURce[<n>]]:CURRent:PROTection:TRIPped?

Syntax

[:SOURce[<*n*>]**]:CURRent:PROTection:TRIPped?**

Description

Queries whether an overcurrent protection (OCP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-

Remarks

- The overcurrent protection (OCP) function disables the output automatically when the actual output current reaches the OCP level.
- If [:SOURce[<n>]] or [<n>] is omitted, the command queries whether an OCP event occurred on the current channel.
- You can also send *:OUTPut:OCP:ALAR?* or *:OUTPut:OCP:QUES?* to query whether an OCP event has occurred on the specified channel.
- You can send *[:SOURce[<n>]]:CURRent:PROTection:CLEar* to clear an OCP event that occurred on the specified channel.

Return Format

The query returns 1 or 0.

Examples

```
:CURR:PROT:TRIP? /*Queries whether an OCP event occurred on the
current channel. The query returns 1.*/
```

4.9.7 [:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]

Syntax

[:SOURce[<*n*>]**]:VOLTage[:LEVel][:IMMediate][:AMPLitude]** {<*voltage*>|<*lim*>|<*amp*>}

[:SOURce[<*n*>]**]:VOLTage[:LEVel][:IMMediate][:AMPLitude]?** [<*lim*>]

Description

Sets or queries the voltage for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-
<voltage>	Real	Please refer to *Table 4.35 Range and default value of overvoltage/overcurrent protection*	
<lim>	Discrete	{MINimum|MAXimum|DEFault}	-
<amp>	Discrete	{UP|DOWN}	-

Remarks

- If [:SOURce[<n>]] or [<n>] is omitted, the command sets the corresponding parameter for the channel currently selected.
- When <voltage> is selected, the command directly sets the voltage for the specified channel. When MINimum, MAXimum, or DEFault is selected, the command sets the minimum, maximum, or default voltage value within the available range for the specified channel. When UP or DOWN is selected, the command steps up or down the voltage by the step size set in *[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]*.
- You can also send *:APPLy* to set the voltage and current for the specified channel.

Return Format

The query returns the voltage of the specified channel, for example, 7.500.

Examples

```
:VOLT 7.5 /*Sets the voltage to 7.5 V for the current channel.*/
:VOLT? /*Queries the voltage for the current channel. The query
returns 7.500.*/
```

4.9.8 [:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]

Syntax

[:SOURce[<*n*>]**]:VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]** {<*voltage*>|<*def*>}

[:SOURce[<*n*>]**]:VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]?** [<*def*>]

Description

Sets or queries the step size of voltage for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-
<voltage>	Real	0 to the maximum voltage value of the specified channel	Please refer to *Remarks*
<def>	Discrete	{DEFault}	-

Remarks

- If [:SOURce<n>] is omitted, the command sets the corresponding parameter of the channel currently selected.
- <voltage> is the step size specified. DEFault is the default value. The default values of <voltage> are shown in the table below:

Channel	Default
CH1	0.001 V
CH2	0.001 V
CH3	0.001 V

- Select UP or DOWN and executes *[:SOURce[<n>]:VOLTage[:LEVel][:IMMediate][:AMPLitude]*, and the instrument will step up or down the voltage by the step size set in this command.

Return Format

The query returns the step size, for example, 0.100.

Examples

```
:VOLT:STEP 0.1 /*Sets the step size of voltage to 0.1 V for the
current channel.*/
:VOLT:STEP? /*Queries the step size of voltage for the current
channel. The query returns 0.100.*/
```

4.9.9 [:SOURce[<n>]]:VOLTage:PROTection:CLEar

Syntax

[:SOURce[<*n*>]:VOLTage:PROTection:CLEar

Description

Clears an OVP event that occurred on the specified channel and enables the output of the corresponding channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-

Remarks

- You can send *[:SOURce[<n>]:VOLTage:PROTection:TRIPped?* to query whether an OVP event occurred on the specified channel.
- Before executing the command, make sure to remove the condition that caused the overvoltage protection on the specified channel (you can either decrease the output voltage to below the OVP level or increase the OVP level to be greater than the output voltage). Execute this command to clear an OVP event that occurred on the specified channel and enable the output of the corresponding channel.
- If [:SOURce[<n>]] or [<n>] is omitted, the command clears an OVP event occurred on the current channel.
- You can send *:OUTPut:OVP:CLEar* command to clear an OVP event that occurred on the specified channel.

Return Format

None.

Examples

```
:VOLT:PROT:TRIP? /*Queries whether an OVP event occurred on the
current channel. The query returns 1.*/
:VOLT:PROT:CLE /*Clears an OVP event that occurred on the current
channel.*/
:VOLT:PROT:TRIP? /*Queries whether an OVP event occurred on the
current channel. The query returns 0.*/
```

4.9.10 [:SOURce[<n>]]:VOLTage:PROTection[:LEVel]

Syntax

[:SOURce[<*n*>]**]:VOLTage:PROTection[:LEVel]** {<*voltage*>|<*lim*>}

[:SOURce[<*n*>]**]:VOLTage:PROTection[:LEVel]?** [<*lim*>]

Description

Sets or queries the overvoltage protection (OVP) level of the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-
<voltage>	Real	Please refer to *Table 4.35 Range and default value of overvoltage/overcurrent protection*	
<lim>	Discrete	{MINimum|MAXimum}	-

Remarks

- When OVP (*[:SOURce[<n>]]:VOLTage:PROTection:STATe*) is enabled, the output is disabled automatically if the actual output voltage reaches the OVP level currently set. You can send *[:SOURce[<n>]]:VOLTage:PROTection:TRIPped?* to query whether an OVP event occurred on the specified channel.
- If [:SOURce[<n>]] or [<n>] is omitted, the command sets the corresponding parameter of the channel currently selected.
- You can also send *:OUTPut:OVP:VALue* to set the OVP level of the specified channel.

Return Format

The query returns the OVP level of the specified channel, for example, 8.800.

Examples

```
:VOLT:PROT 8.8 /*Sets the OVP level to 8.8 V for the selected
channel.*/
:VOLT:PROT? /*Queries the OVP level for the selected channel. The
query returns 8.800.*/
```

4.9.11 [:SOURce[<n>]]:VOLTage:PROTection:STATe

Syntax

[:SOURce[<*n*>]]:VOLTage:PROTection:STATe <*bool*>

[:SOURce[<*n*>]]:VOLTage:PROTection:STATe?

Description

Sets or queries the on/off state of the overvoltage protection (OVP) function of the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-
<bool>	Bool	{0|1|ON|OFF}	0

Remarks

- When OVP (*[:SOURce[<n>]]:VOLTage:PROTection:STATe*) is enabled, the output is disabled automatically if the actual output voltage reaches the OVP level currently set. You can send *[:SOURce[<n>]]:VOLTage:PROTection:TRIPped?* to query whether an OVP event occurred on the specified channel.
- If [:SOURce[<n>]] or [<n>] is omitted, the command sets the corresponding parameter of the channel currently selected.
- You can also send *:OUTPut:OVP[:STATe]* to enable or disable the OVP function of the specified channel.
- You can also send *[:SOURce[<n>]]:VOLTage:PROTection[:LEVel]* to query the OVP level of the specified channel.

Return Format

The query returns 1 or 0.

Examples

```
:VOLT:PROT:STAT ON /*Enables the OVP function for the current
channel.*/
:VOLT:PROT:STAT? /*Queries the on/off state of the OVP function of
the current channel. The query returns 1.*/
```

4.9.12 [:SOURce[<n>]]:VOLTage:PROTection:TRIPped?

Syntax

[:SOURce[<*n*>]]:VOLTage:PROTection:TRIPped?

Description

Queries whether an overvoltage protection (OVP) event occurred on the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-

Remarks

- The overvoltage protection (OVP) function disables the output automatically when the actual output voltage reaches the OVP level.
- If [:SOURce[<n>]] or [<n>] is omitted, the command queries whether an OVP event occurred on the current channel.
- You can also send *:OUTPut:OVP:ALAR?* or *:OUTPut:OVP:QUES?* to query whether an OVP event has occurred on the specified channel.
- You can send *[:SOURce[<n>]]:VOLTage:PROTection:CLEar* to clear an OVP event that occurred on the specified channel.

Return Format

The query returns 1 or 0.

Examples

```
:VOLT:PROT:TRIP? /*Queries whether an OVP event occurred on the
current channel. The query returns 1.*/
```

4.10 :STATus Commands

:STATus commands are used to set and query the Questionable Status register and Operation Status register.

4.10.1 :STATus:OPERation:CONDition?

Syntax

:STATus:OPERation:CONDition?

Description

Queries the value of the condition register of the Operation Status register.

Parameter

None.

Remarks

None.

Return Format

If the instrument is in calibration, the query returns +1. Otherwise, the query returns +0.

Examples

```
:STATus:OPERation:CONDition? /*Queries the value of the condition
register of the Operation Status register. The query returns +0.*/
```

4.10.2 :STATus:OPERation:ENABle

Syntax

:STATus:OPERation:ENABle <*value*>

:STATus:OPERation:ENABle?

Description

Enables the bits in the enable register of the Operation Status register.

Queries the enabled bits in the enable register of the Operation Status register.

Parameter

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

Remarks

- The <value> is a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Operation Status register.
- When <value> is set to 0, executing this command will clear the enable register of the Operation Status register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Operation Status register. For example, the query might return +16.

Examples

```
STATus:OPERation:ENABle 16 /*Enables bit4 in the enable register.*/
:STATus:OPERation:ENABle? /*Queries the enabled bits. The query
returns +16.*/
```

4.10.3 :STATus:OPERation[:EVENt]?

Syntax

:STATus:OPERation[:EVENt]?

Description

Queries the value of the event register of the Operation Status register.

Parameter

None.

Remarks

Executes this command and the query returns a decimal value (corresponding to the binary-weighted sum of all bits set in the register) and clear the status of the register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits in the register. For example, the query might return +17.

Examples

```
:STAT:OPER? /*Queries the value of the event register of the
Operation Status register.*/
```

4.10.4 :STATus:PRESet

Syntax

:STATus:PRESet

Description

Sets the enable registers of the Questionable Status register to their power-on defaults.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
:STATus:PRESet /*Sets the enable registers of the Questionable
Status register to their power-on defaults.*/
```

4.10.5 :STATus:QUEStionable:ENABle

Syntax

:STATus:QUEStionable:ENABle <*enable value*>

:STATus:QUEStionable:ENABle?

Description

Enables the bits in the enable register of the Questionable Status register.

Queries the enabled bits in the enable register of the Questionable Status register.

Parameter

Name	Type	Range	Default
<enable value>	Integer	Refer to *Remarks*	-

Remarks

- The <enable value> is a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Questionable Status register.
- Enable the bits in the enable register of the Questionable Status register and the system will report the state of the corresponding bit to the Status Byte register.
- When <enable value> is set to 0, executing this command will clear the enable register of the Questionable Status register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Questionable Status register. For example, the query might return +17.

Examples

```
:STAT:QUES:ENAB 17 /*Enables bit0 and bit4 in the enable register
of the Questionable Status register.*/
:STAT:QUES:ENAB? /*Queries the enabled bits in the enable register
of the Questionable Status register. The query returns +17.*/
```

4.10.6 :STATus:QUEStionable[:EVENt]?

Syntax

:STATus:QUEStionable[:EVENt]?

Description

Queries the enable register of the Questionable Status register.

Parameter

None.

Remarks

- Executes this command and the query returns a decimal value (corresponding to the binary-weighted sum of all bits set in the register) and clear the status of the register.
- The bits in the event register of the Questionable Status register are latched and reading the register will clear it. You can also send **CLS* to clear the register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits set in the event register of the Questionable Status register. For example, the query may return +17.

Examples

```
:STAT:QUES? /*Queries the enable register of the Questionable
Status register. The query returns +17.*/
```

4.10.7 :STATus:QUEStionable:INSTrument:ENABle

Syntax

:STATus:QUEStionable:INSTrument:ENABle <*enable value*>

:STATus:QUEStionable:INSTrument:ENABle?

Description

Enables the bits in the enable register of the Questionable Status register.

Queries the enabled bits in the enable register of the Questionable Status register.

Parameter

Name	Type	Range	Default
<enable value>	Integer	Refer to *Remarks*	-

Remarks

- The <enable value> is a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the Questionable Status register.
- Enable the bits in the enable register of the Questionable Status register and the system will report the state of the corresponding bit to the Status Byte register.
- When <enable value> is set to 0, executing this command will clear the enable register of the Questionable Status register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits enabled in the enable register of the Questionable Status register. For example, the query might return +14.

Examples

```
:STAT:QUES:INST:ENAB 14 /*Enables bit1, bit2, and bit3 (INST(n)
event summary), channel (n) event summary bit; (n)=1, 2, or 3) in
the enable register of the Questionable Status register.*/
:STAT:QUES:INST:ENAB? /*Queries the enabled bits in the enable
register of the Questionable Status register. The query returns
+14.*/
```

4.10.8 :STATus:QUEStionable:INSTrument[:EVENt]?

Syntax

`:STATus:QUEStionable:INSTrument[:EVENt]?`

Description

Queries the event register of the channel Questionable Status register.

Parameter

None.

Remarks

- Executes this command and the query returns a decimal value (corresponding to the binary-weighted sum of all bits set in the register) and clear the status of the register.
- The bits in the event register of the channel Questionable Status register are latched and reading the register will clear it. You can also send **CLS* to clear the register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits set in the event register of the channel Questionable Status register. For example, the query may return +10.

Examples

```
:STAT:QUES:INST? /*Queries the event register of the channel
Questionable Status register. The query returns +10.*/
```

4.10.9 :STATus:QUEStionable:INSTrument:ISUMmary[<n>]:CONDition?

Syntax

:STATus:QUEStionable:INSTrument:ISUMmary[<*n*>]**:CONDition?**

Description

Sets or queries the output mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Integer	{1|2|3}	-

Remarks

- If [<n>] is omitted, the command queries the output mode of the current channel.
- Execute the command and the query returns +0, +1, +2, or +3, as shown in the table below.

Returned Value	Description
+0	The output is off.
+1	The output is in CC (constant current) mode.
+2	The output is in CV (constant voltage) mode.
+3	The output is in UR (unregulated) mode.

Return Format

The query returns +0, +1, +2, or +3.

Examples

```
:STAT:QUES:INST:ISUM1:COND? /*Queries the output mode of CH1. The
query returns +1.*/
```

4.10.10 :STATus:QUEStionable:INSTrument:ISUMmary[<n>]:ENABle

Syntax

:STATus:QUEStionable:INSTrument:ISUMmary[<*n*>]**:ENABle** <*enable value*>

:STATus:QUEStionable:INSTrument:ISUMmary[<*n*>]**:ENABle?**

Description

Enables the bits in the enable register of the specified channel Questionable Status SUMMARY register.

Queries the enabled bits in the enable register of the specified channel Questionable Status SUMMARY register.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3}	-
<enable value>	Integer	Refer to *Remarks*	-

Remarks

- This command is available for multi-channel models only. Multi-channel models have multiple channel Questionable Status SUMMARY registers. The particular channel is specified by a numeric value ([<n>]=1, 2, or 3). If [<n>] is omitted, the command queries the enable register of the current channel Questionable Status SUMMARY register.
- The <enable value> is a decimal value, which corresponds to the binary-weighted sum of the bits to be enabled in the enable register of the channel Questionable Status SUMMARY register.
- Enable the bits in the enable register of the specified channel Questionable Status SUMMARY register and the system will report the state of the corresponding bit to the channel Questionable Status register.
- When <enable value> is set to 0, executing this command will clear the enable register of the channel Questionable Status SUMMARY register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits enabled in the enable register of the channel Questionable Status SUMMARY register. For example, the query might return +9.

Examples

```
:STAT:QUES:INST:ISUM1:ENAB 9 /*Queries bit0 (Voltage, CC mode) and
bit3 (OCP, overcurrent protection) in the enable register of the
Questionable Status SUMMARY register for CH1.*/
:STAT:QUES:INST:ISUM1:ENAB? /*Queries the enabled bits in the
enable register of the Questionable Status SUMMARY register for
CH1. The query returns +9.*/
```

4.10.11 :STATus:QUEStionable:INSTrument:ISUMmary[<n>][:EVENt]?

Syntax

:STATus:QUEStionable:INSTrument:ISUMmary[<*n*>]**[:EVENt]?**

Description

Queries the value of the event register of the specified channel Questionable Status SUMMARY register.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3}	-

Remarks

- Multi-channel models have multiple channel Questionable Status SUMMARY registers. The particular channel is specified by a numeric value ([<n>]=1, 2 3). If [<n>] is omitted, the command queries the enable register of the current channel Questionable Status SUMMARY register.
- Executes this command and the query returns a decimal value (corresponding to the binary-weighted sum of all bits set in the register) and clear the status of the register.
- This event register latches all bits. Reading the register clears it. You can also send **CLS* to clear the register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of the bits enabled in the enable register of the channel Questionable Status SUMMARY register. For example, the query might return +1.

Examples

```
:STAT:QUES:INST:ISUM1? /*Queries the value of the event register of
the Questionable Status SUMMARY register for CH1. The query returns
+1.*/
```

4.11 :SYSTem Commands

:SYSTem commands are used to perform system setting, output setting, and interface parameter setting.

4.11.1 :SYSTem:BEEPer:IMMediate

Syntax

`:SYSTem:BEEPer:IMMediate`

Description

Issues a single beep immediately.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

None.

4.11.2 :SYSTem:BEEPer[:STATe]

Syntax

`:SYSTem:BEEPer[:STATe]` <*bool*>

`:SYSTem:BEEPer[:STATe]?`

Description

Sets or queries the on/off state of the beeper.

Parameter

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

Remarks

When beeper is enabled, the instrument enables the click sound when the touch screen is used or the front-panel keys and knob are used, or enables the beep sound when an error is generated from the remote control.

Return Format

The query returns 0 or 1.

Examples

```
:SYST:BEEP ON /*Turns on the beeper.*/
:SYST:BEEP? /*Queries the on/off state of the beeper. The query
returns 1.*/
```

4.11.3 :SYSTem:BRIGhtness

Syntax

:SYSTem:BRIGhtness {<*brightness*>|<*lim*>}

:SYSTem:BRIGhtness? [<*lim*>]

Description

Sets or queries the brightness of the LCD screen.

Parameter

Name	Type	Range	Default
<brightness>	Integer	1 to 100	50 (factory)
<lim>	Discrete	{MINimum|MAXimum}	-

Remarks

MINimum and MAXimum indicate the minimum and maximum brightness value available respectively.

Return Format

The query returns an integer, for example, 60.

Examples

```
:SYST:BRIG 60 /*Sets the screen brightness to 60%.*/
:SYST:BRIG? /*Queries the screen brightness. The query returns 60.*/
```

4.11.4 :SYSTem:COMMunicate:LAN

4.11.4.1 :SYSTem:COMMunicate:LAN:APPLy

Syntax

:SYSTem:COMMunicate:LAN:APPLy

Description

Applies the network parameters currently set.

Parameter

None.

Remarks

The new setting is valid only when this command is executed after the LAN parameters are set.

Return Format

None.

Examples

```
:SYST:COMM:LAN:APPL /*Applies the network parameters currently
set.*/
```

4.11.4.2 :SYSTem:COMMunicate:LAN:AUTOip[:STATe]

Syntax

:SYSTem:COMMunicate:LAN:AUTOip[:STATe] <*bool*>

:SYSTem:COMMunicate:LAN:AUTOip[:STATe]?

Description

Sets or queries the on/off state of Auto IP.

Parameter

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

Remarks

- Before using the LAN interface, please use the network cable to connect the instrument to your PC or to the network of the PC.
- The instrument provides three IP configuration modes: DHCP, Auto IP, and Manual IP.

- When operating in Auto IP mode, the instrument automatically acquires the IP address from 169.254.0.1 to 169.254.255.254 and subnet mask 255.255.0.0 according to the current network configuration.
- When all the three configuration modes are set to "On", the priority of parameter configuration is "DHCP", "Auto IP", and "Manual IP". Therefore, to use the Auto IP configuration mode, "DHCP (*:SYSTem:COMMunicate:LAN:DHCP[:STATe]*)" should be set to "OFF".
- The three IP configuration modes cannot be set to "OFF" at the same time.
- After sending the command, execute *:SYSTem:COMMunicate:LAN:APPLy* to apply the network parameters currently set.

Return Format

The query returns 1 or 0.

Examples

```
:SYST:COMM:LAN:AUTO ON /*Enables the Auto IP mode.*/
:SYST:COMM:LAN:AUTO? /*Queries the on/off state of Auto IP. The
query returns 1.*/
```

4.11.4.3 :SYSTem:COMMunicate:LAN:DHCP[:STATe]

Syntax

:SYSTem:COMMunicate:LAN:DHCP[:STATe] <*bool*>

:SYSTem:COMMunicate:LAN:DHCP[:STATe]?

Description

Sets or queries the on/off state of DHCP.

Parameter

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

Remarks

- In DHCP mode, the DHCP server in the current network assigns network parameters (such as the IP address) to the instrument.

- When all the three configuration modes are set to "On", the priority of parameter configuration is "DHCP", "Auto IP", and "Manual IP".
- The three IP configuration modes cannot be set to "OFF" at the same time.
- After sending the command, execute *:SYSTem:COMMunicate:LAN:APPLy* to apply the network parameters currently set.

Return Format

The query returns 1 or 0.

Examples

```
:SYST:COMM:LAN:DHCP ON /*Enables the DHCP mode.*/
:SYST:COMM:LAN:DHCP? /*Queries the on/off state of DHCP. The query returns 1.*/
```

4.11.4.4 :SYSTem:COMMunicate:LAN:DNS

Syntax

:SYSTem:COMMunicate:LAN:DNS <*dns*>

:SYSTem:COMMunicate:LAN:DNS?

Description

Sets or queries the Domain Name Service (DNS) address.

Parameter

Name	Type	Range	Default
<dns>	ASCII string	Refer to *Remarks*	-

Remarks

- The command is valid only when Manual IP is enabled (*:SYSTem:COMMunicate:LAN:MANualip[:STATe]*).
- The format of <dns> is nnn.nnn.nnn.nnn; wherein, the first nnn ranges from 1 to 223 (except 127), and the other three range from 0 to 255.
- It is recommended that you acquire a valid address from your network administrator.
- After sending the command, execute *:SYSTem:COMMunicate:LAN:APPLy* to apply the network parameters currently set.

Return Format

The query returns the DNS address, for example, 172.16.3.2.

Examples

```
:SYST:COMM:LAN:DNS 172.16.3.2 /*Sets the DNS address to
172.16.3.2.*/
:SYST:COMM:LAN:DNS? /*Queries the current DNS address. The query
returns 172.16.3.2.*/
```

4.11.4.5 :SYSTem:COMMunicate:LAN:IPADdress

Syntax

:SYSTem:COMMunicate:LAN:IPADdress <*ip*>

:SYSTem:COMMunicate:LAN:IPADdress?

Description

Sets or queries the IP address.

Parameter

Name	Type	Range	Default
<ip>	ASCII string	Refer to *Remarks*	-

Remarks

- The command is valid only when Manual IP is enabled (*:SYSTem:COMMunicate:LAN:MANualip[:STATe]*).
- The format of <ip> is nnn.nnn.nnn.nnn; wherein, the first nnn ranges from 1 to 223 (except 127), and the other three range from 0 to 255.
- It is recommended that you acquire a valid address from your network administrator.
- After sending the command, execute *:SYSTem:COMMunicate:LAN:APPLy* to apply the network parameters currently set.

Return Format

The query returns the IP address, for example, 172.16.3.128.

Examples

```
:SYST:COMM:LAN:DNS 172.16.3.128 /*Sets the IP address to
172.16.3.128.*/
:SYST:COMM:LAN:DNS? /*Queries the current IP address. The query
returns 172.16.3.128.*/
```

4.11.4.6 :SYSTem:COMMunicate:LAN:GATEway

Syntax

:SYSTem:COMMunicate:LAN:GATEway <*gateway*>

:SYSTem:COMMunicate:LAN:GATEway?

Description

Sets or queries the default gateway.

Parameter

Name	Type	Range	Default
<gateway>	ASCII string	Refer to *Remarks*	-

Remarks

- The command is valid only when Manual IP is enabled (*:SYSTem:COMMunicate:LAN:MANualip[:STATe]*).
- The format of <gateway> is nnn.nnn.nnn.nnn; wherein, the first nnn ranges from 1 to 223 (except 127), and the other three range from 0 to 255.
- It is recommended that you acquire a valid gateway address from your network administrator.
- After sending the command, you must execute *:SYSTem:COMMunicate:LAN:APPLy* to apply the network parameters currently set.

Return Format

The query returns the default gateway, for example, 172.16.3.1.

Examples

```
:SYST:COMM:LAN:GATE 172.16.3.1 /*Sets the default gateway to
172.16.3.1.*/
:SYST:COMM:LAN:GATE? /*Queries the current default gateway. The
query returns 172.16.3.1.*/
```

4.11.4.7 :SYSTem:COMMunicate:LAN:MAC?

Syntax

:SYSTem:COMMunicate:LAN:MAC?

Description

Queries the MAC address.

Parameter

None.

Remarks

The MAC (Media Access Control) address, also called the hardware address, is used to define the location of the network device. For a power supply, the MAC address is always unique, and is usually used to recognize the instrument when assigning IP address to the instrument. The MAC address (48 bits, namely 6 bytes) is usually expressed in hexadecimal form, for example, 00-2A-A0-AA-E0-56.

Return Format

The query returns the MAC address, for example, 00-2A-A0-AA-E0-56.

Examples

None.

4.11.4.8 :SYSTem:COMMunicate:LAN:MANualip[:STATe]

Syntax

`:SYSTem:COMMunicate:LAN:MANualip[:STATe]` <*bool*>

`:SYSTem:COMMunicate:LAN:MANualip[:STATe]?`

Description

Sets or queries the on/off state of Manual IP.

Parameter

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

Remarks

- In Manual IP mode, you can define the network parameters (such as the IP address).
- When all the three configuration modes are set to "ON", the priority of parameter configuration is "DHCP", "Auto IP", and "Manual IP". Therefore, to use the Manual IP configuration mode, “DHCP

(*:SYSTem:COMMunicate:LAN:DHCP[:STATe]*)" and "Auto IP (*:SYSTem:COMMunicate:LAN:AUTOip[:STATe]*)" should be set to "OFF".

- The three IP configuration modes cannot be set to "OFF" at the same time.
- After sending the command, execute *:SYSTem:COMMunicate:LAN:APPLy* to apply the network parameters currently set.

Return Format

The query returns 1 or 0.

Examples

```
:SYST:COMM:LAN:MAN ON /*Enables the Manual IP mode.*/
:SYST:COMM:LAN:MAN? /*Queries the on/off state of Manual IP. The
query returns 1.*/
```

4.11.4.9 :SYSTem:COMMunicate:LAN:SMASk

Syntax

:SYSTem:COMMunicate:LAN:SMASk <*submask*>

:SYSTem:COMMunicate:LAN:SMASk?

Description

Sets or queries the subnet mask.

Parameter

Name	Type	Range	Default
<submask>	ASCII string	Refer to *Remarks*	-

Remarks

- The command is valid only when Manual IP is enabled (*:SYSTem:COMMunicate:LAN:MANualip[:STATe]*).
- The format of <submask> is nnn.nnn.nnn.nnn; wherein, the nnn ranges from 0 to 255.
- It is recommended that you acquire a valid subnet mask from your network administrator.
- After sending the command, you must execute *:SYSTem:COMMunicate:LAN:APPLy* to apply the network parameters currently set.

Return Format

The query returns the subnet mask, for example, 255.255.255.0.

Examples

```
:SYST:COMM:LAN:SMAS 255.255.255.0 /*Sets the subnet mask to
255.255.255.0.*/
:SYST:COMM:LAN:SMAS? /*Queries the subnet mask. The query returns
255.255.255.0.*/
```

4.11.5 :SYSTem:COMMunicate:RLSTate

Syntax

:SYSTem:COMMunicate:RLSTate <*mode*>

:SYSTem:COMMunicate:RLSTate?

Description

Sets the power supply to remote, local mode, or remote lock mode.

Queries the current operation mode.

Parameter

Name	Type	Range	Default
<mode>	Discrete	{LOCal|REMote|RWLock}	LOCal

Remarks

- **LOCal:** Local mode.
- **REMote:** Remote mode. The touch screen and all front-panel keys are disabled except for the front-panel output on/off keys and 

key. At this point, you can press the output on/off key only to disable channel output.
- **RWLock:** Remote lock mode. The touch screen and all front-panel keys are disabled except for the front-panel output on/off keys. At this point, you can press the output on/off key only to disable channel output. You can only use the specified command to disable the remote lock mode.

Return Format

The query returns LOCal, REMote, or RWLock.

Examples

```
:SYSTem:COMMunicate:RLSTate REMote /*Sets the power supply to
remote mode.*/
:SYSTem:COMMunicate:RLSTate? /*Queries the operation mode. The
query returns REMote.*/
```

4.11.6 :SYSTem:COMMunicate:RS232

4.11.6.1 :SYSTem:COMMunicate:RS232:BAUD

Syntax

:SYSTem:COMMunicate:RS232:BAUD <*baud*>

:SYSTem:COMMunicate:RS232:BAUD?

Description

Sets or queries the baud rate of the RS232 interface. The unit is Baud.

Parameter

Name	Type	Range	Default
<baud>	Discrete	{9600|19200|38400|57600|115200}	9600

Remarks

Connect the RS232 interface to your PC or data terminal equipment (DTE) using RS232 cable and set the interface parameters (baud rate, parity bit, etc.) that match the PC or terminal equipment.

Return Format

The query returns the baud rate, for example, 19200.

Examples

```
:SYST:COMM:RS232:BAUD 19200 /*Sets the baud rate of the RS232
interface to 19200.*/
:SYST:COMM:RS232:BAUD? /*Queries the baud rate of the RS232
interface. The query returns 19200.*/
```

4.11.6.2 :SYSTem:COMMunicate:RS232:DBIT

Syntax

:SYSTem:COMMunicate:RS232:DBIT <*databit*>

:SYSTem:COMMunicate:RS232:DBIT?

Description

Sets or queries the data bit of the RS232 interface.

Parameter

Name	Type	Range	Default
<databit>	Discrete	{7|8}	8

Remarks

None.

Return Format

The query returns 7 or 8.

Examples

```
:SYST:COMM:RS232:DBIT 8 /*Sets the data bit of the RS232 interface
to 8.*/
:SYST:COMM:RS232:DBIT? /*Queries the data bit of the RS232
interface. The query returns 8.*/
```

4.11.6.3 :SYSTem:COMMunicate:RS232:PBIT

Syntax

:SYSTem:COMMunicate:RS232:PBIT <*bit*>

:SYSTem:COMMunicate:RS232:PBIT?

Description

Sets or queries the parity check of RS232.

Parameter

Name	Type	Range	Default
<bit>	Discrete	{NONE|ODD|EVEN}	NONE

Remarks

Selecting NONE, ODD, or EVEN indicates setting the ParityBit to "None", "Odd", or "Even".

Return Format

The query returns NONE, ODD, or EVEN.

Examples

```
:SYST:COMM:RS232:PBIT ODD /*Sets the parity check to Odd.*/
:SYST:COMM:RS232:PBIT? /*Queries the parity check. The query
returns ODD.*/
```

4.11.6.4 :SYSTem:COMMunicate:RS232:SBIT

Syntax

:SYSTem:COMMunicate:RS232:SBIT <*n*>

:SYSTem:COMMunicate:RS232:SBIT?

Description

Sets or queries the stop bit of RS232.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2}	1

Remarks

None.

Return Format

The query returns 1 or 2.

Examples

```
:SYST:COMM:RS232:SBIT 2 /*Sets the stop bit to 2.*/
:SYST:COMM:RS232:SBIT? /*Queries the stop bit. The query returns
2.*/
```

4.11.7 :SYSTem:ERRor[:NEXT]?

Syntax

:SYSTem:ERRor[:NEXT]?

Description

Queries and removes errors from the error queue.

Parameter

None.

Remarks

The power supply beeps once each time an error is detected. If more than 20 errors have occurred, the last error stored in the queue (the most recent error) is replaced with -350, "Queue overflow". No additional errors are stored until you remove errors from the queue.

Errors are retrieved in first-in-first-out (FIFO) order.

The error queue is cleared when power has been off or after **CLS* has been executed. The **RST* command does not clear the error queue.

Return Format

The query returns the name and content of the error message, for example, -113,"Undefined header; keyword cannot be found" . If no error has occurred, the query returns 0, "No error" .

Examples

```
None.
```

4.11.8 :SYSTem:KLOCk:STATe

Syntax

:SYSTem:KLOCk:STATe <*bool*>

:SYSTem:KLOCk:STATe?

Description

Sets or queries the on/off state of remote lock.

Parameter

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

Remarks

- When the instrument operates in remote lock mode, all the keys on the front panel are disabled except for the output on/off key of each channel  ,  , and the power switch key  . At this point, you can press the output on/off key only to disable channel output.
- You can also send *:SYSTem:RWLock* to enable or disable remote lock.

Return Format

The query returns 1 or 0.

Examples

```
:SYST:KLOC:STAT ON /*Enables the remote lock.*/
:SYST:KLOC:STAT? /*Queries the on/off state of remote lock. The
query returns 1.*/
```

4.11.9 :SYSTem:LANGuage:TYPE

Syntax

:SYSTem:LANGuage:TYPE <*type*>

:SYSTem:LANGuage:TYPE?

Description

Sets or queries the system language.

Parameter

Name	Type	Range	Default
<type>	Discrete	{EN|CH|DE|ES|FR}	-

Remarks

None.

Return Format

The query returns ENGLISH, CHINESE, GERMAN, SPANISH, or FRENCH.

Examples

```
:SYST:LANG:TYPE EN /*Sets the system language to English.*/
:SYST:LANG:TYPE? /*Queries the system language. The query returns
ENGLISH.*/
```

4.11.10 :SYSTem:LOCal

Syntax

:SYSTem:LOCal

Description

Returns the power supply from remote mode to local mode.

Parameter

None.

Remarks

- When the instrument operates in remote mode, all the keys on the front panel are disabled except for the output on/off key of each channel  ,  , power switch key  , and  . At this point, you can press the output on/off key

only to disable channel output. The command returns the power supply from remote control to local operation mode. At this point, all the front-panel keys can be used.

- You can send *:SYSTem:REMote* to return the power supply from local mode to remote mode.

Return Format

None.

Examples

```
None.
```

4.11.11 :SYSTem:POWEron

Syntax

:SYSTem:POWEron <*poweron*>

:SYSTem:POWEron?

Description

Sets or queries the power-on setting.

Parameter

Name	Type	Range	Default
<poweron>	Discrete	{DEFault|LAST}	DEFault

Remarks

- "DEFault" indicates using the factory default setting.
- "Last" indicates using the system configuration before the last power-off.

Return Format

The query returns DEF or LAST.

Examples

```
:SYST:POWE LAST /*Sets the instrument to use the system
configuration before the last power-off at power-on.*/
:SYST:POWE? /*Queries the instrument configuration to be used at
power-on. The query returns LAST.*/
```

4.11.12 :SYSTem:PRINt?

Syntax

```
:SYSTem:PRINt?
```

Description

Queries the byte stream of the current screen image.

Parameter

None.

Remarks

None.

Return Format

The query returns the hexadecimal string of the screen image in bitmap (*.bmp) format.

Examples

```
None.
```

4.11.13 :SYSTem:REMote

Syntax

```
:SYSTem:REMote
```

Description

Returns the power supply from local mode to remote mode.

Parameter

None.

Remarks

- When the instrument operates in local mode, all front-panel keys can be used. Execute this command to return the power supply from local mode to remote mode. At this point, all the keys on the front panel cannot be used except for the output on/off key of each channel 

, 

, power switch key  , and 

.
- You can send *:SYSTem:LOCal* return the power supply from remote mode to local mode.

Return Format

None.

Examples

```
None.
```

4.11.14 :SYSTem:RWLock

Syntax

:SYSTem:RWLock[:STATe] [<*bool*>]

Description

Enables or disables remote lock.

Parameter

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

Remarks

- When the instrument operates in remote lock mode, all the keys on the front panel cannot be used except for the output on/off key of each channel  ,  , and the power switch key  . At this point, you can press the output on/off key only to disable channel output.
- If <bool> is omitted, the command enables the remote lock.
- You can also send *:SYSTem:KLOCk:STATe* to enable or disable remote lock.

Return Format

The query returns 1 or 0.

Examples

```
:SYST:RWL ON /*Enables the remote lock.*/
```

4.11.15 :SYSTem:SAMPling

Syntax

:SYSTem:SAMPling <*mode*>

:SYSTem:SAMPling?

Description

Sets or queries the sampling mode.

Parameter

Name	Type	Range	Default
<mode>	Discrete	{AUTO|HIGH|LOW}	AUTO

Remarks

- DP2000 series power supply provides three current sampling modes for CH1 and CH2. You can select "High Curr", "Low Curr", or "AUTO", and the default is "AUTO".
  - **HIGH:** The power supply uses the high range current mode.
  - **LOW:** The maximum current that can be programmed is 0.5 A and the maximum current that can be read is 11 mA. When 11 mA is reached, the output current is displayed as "-.----" in the interface. When the power supply outputs current within 11 mA, it uses the low range current mode.
  - **AUTO:** When the power supply outputs current higher than or equal to 11 mA, it uses the high range current mode; when the power supply outputs current lower than 11 mA, it automatically switches to the low range current measurement mode.
- The three sampling modes are only available for CH1 and CH2. CH3 has no low current sampling mode, and hence cannot switch sampling mode.
- When the instrument operates in parallel mode (internal), the sampling mode is fixed to high range current mode. If you want to use external parallel connection, you need to set the sampling mode to high current mode manually.

Return Format

The query returns AUTO, HIGH, or LOW.

Examples

```
:SYSTem:SAMPling HIGH /*Sets the sampling mode to high range mode.*/
:SYSTem:SAMPling? /*Queries the sampling mode. The query returns
HIGH.*/
```

4.11.16 :SYSTem:SAVer

Syntax

:SYSTem:SAVer <*bool*>

:SYSTem:SAVer?

Description

Sets or queries the on/off state of the screen saver function.

Parameter

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

Remarks

None.

Return Format

The query returns 1 or 0.

Examples

```
:SYST:SAV ON /*Enables the screen saver function.*/
:SYST:SAV? /*Queries the on/off state of the screen saver function.
The query returns 1.*/
```

4.11.17 :SYSTem:SENSe

Syntax

:SYSTem:SENSe <*source*>,<*bool*>

:SYSTem:SENSe? <*source*>

Description

Sets or queries the on/off state of the Sense function for the specified channel.

Parameter

Name	Type	Range	Default
<source>	Discrete	{CH1|CH2|CH3|ALL}	-
<bool>	Bool	{1|0|ON|OFF}	OFF

Remarks

When the output current of the power supply is large, in order to ensure that the load acquires the correct voltage drop, this series power supply provides the Sense (remote sense) working mode. In this mode, the instrument detects the voltage at the load terminal instead of the voltage at the output terminal of the power supply, so the instrument can automatically compensate for the voltage drop caused by the load lead.

Return Format

The query returns 1 or 0.

Examples

```
:SYSTem:SENS CH1,ON /*Enables the Sense function of CH1./
:SYSTem:SENS? CH1 /*Queries the on/off state of the Sense function
for CH1. The query returns 1.*/
```

4.11.18 :SYSTem:SYNC[:STATe]

Syntax

`:SYSTem:SYNC[:STATe]` <*bool*>

`:SYSTem:SYNC[:STATe]?`

Description

Sets or queries the state of the on/off sync function.

Parameter

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

Remarks

- For CH1 and CH2, the on/off state of one channel will change accordingly when that of the other is modified.
- The command is valid only when the tracking function (*:OUTPut:TRACk[:STATe]*) is enabled.

Return Format

The query returns 1 or 0.

Examples

```
:SYST:SYNC ON /*Enables the on/off sync function.*/
:SYST:SYNC? /*Queries the state of the on/off sync function. The
query returns 1.*/
```

4.11.19 :SYSTem:TMODe

Syntax

:SYSTem:TMODe <*trackmode*>

:SYSTem:TMODe?

Description

Sets or queries the on/off state of tracking function.

Parameter

Name	Type	Range	Default
<trackmode>	Discrete	{SYNC|INDE}	INDE

Remarks

The tracking function is available for the specified channels (CH1 and CH2). You can select the track mode as required. This command functions the same as *:OUTPut:TRACk[:STATe]*.

- **SYNC:** Enables the track mode. For the two channels (from a single power supply) that support this mode, changes made on one channel (including voltage/current setting value, OVP/OCP level, and on/off status) is applied to the other channel.
- **INDE:** Disables the track mode. For the two channels (from a single power supply) that support this mode, changes made on one channel will not affect the other.

Return Format

The query returns SYNCHRONOUS or INDEPENDENT.

Examples

```
:SYST:TMOD SYNC /*Enables the track mode.*/
:SYST:TMOD? /*Queries the on/off status of the track mode. The
query returns SYNCHRONOUS.*/
```

4.11.20 :SYSTem:TLOCk

Syntax

:SYSTem:TLOCk <*bool*>

:SYSTem:TLOCk?

Description

Locks or unlocks the touch screen; queries whether the touch screen is locked.

Parameter

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

Remarks

The touch screen cannot be used.

Return Format

The query returns 1 or 0.

Examples

```
:SYSTem:TLOCk ON /*Locks the touch screen.*/
:SYSTem:TLOCk? /*Queries whether the touch screen is locked. The
query returns 1.*/
```

4.11.21 :SYSTem:VERSion?

Syntax

`:SYSTem:VERSion?`

Description

Queries the present SCPI version of the system.

Parameter

None.

Remarks

None.

Return Format

The query returns a string in the form of YYYY.V where "YYYY" represent the year of the version, and the "V" represents the current version number of the SCPI. For example, the query may return 1999.0.

Examples

```
:SYST: VERS? /*Queries the present SCPI version of the system. The
query returns 1999.0.*/
```

4.12 :TIMEr Commands

:TIMEr commands are used to set the parameters and on/off state of the arbitrary waveform generator.

4.12.1 :TIMEr:CYCLEs

Syntax

:TIMEr:CYCLEs <*cycle*>[,<*value*>]

:TIMEr:CYCLEs?

Description

Sets or queries the repetition cycle for the channel currently selected.

Parameter

Name	Type	Range	Default
<cycle>	Discrete	{N|I}	N
<value>	Integer	1 to 99999	1

Remarks

- Repetition cycle refers to the number of cycles that the instrument performs timing output according to the preset voltage and current values. You can set the number of cycles to infinite (I) or a specified value (N,<value>).
- The total number of groups in timing output = the number of output groups × repetition cycle; wherein, you can send *:TIMEr:GROUPs:NUM?* to query the number of output groups.
- The power supply terminates the timer function when the total number of groups is complete. At this point, the state of the power supply is decided by the setting in *:TIMEr:ENDState*.

Return Format

The query returns I or N,<value>, for example, N,20.

Examples

```
:TIME:CYCLE N,20 /*Sets the repetition cycle to 20.*/
:TIME:CYCLE? /*Queries the repetition cycle. The query returns
N,20.*/
```

4.12.2 :TIMEr:CHANNel

Syntax

:TIMEr:CHANNel <*ch*>

:TIMEr:CHANNel?

Description

Sets or queries the channel currently edited.

Parameter

Name	Type	Range	Default
<ch>	Discrete	{CH1|CH2|CH3}	-

Remarks

None.

Return Format

The query returns CH1, CH2, or CH3.

Examples

```
:TIMEr:CHANNel CH2 /*Sets the channel currently edited to CH2.*/
:TIMEr:CHANNel? /*Queries the channel currently edited. The query
returns CH2.*/
```

4.12.3 :TIMEr:ENDState

Syntax

:TIMEr:ENDState <*end*>

:TIMEr:ENDState?

Description

Sets or queries the end state of the generator.

Parameter

Name	Type	Range	Default
<end>	Discrete	{OFF|LAST}	OFF

Remarks

"End state" refers to the state of the instrument after it has completed outputting groups of voltage/current when the number of cycles is finite.

- **OFF:** the instrument shuts down output automatically once output is completed.
- **LAST:** the instrument stays in the output state of the last group after the output is completed.

The total number of groups in timing output = the number of output groups × the number of cycles; wherein, you can send *:TIMEr:GROUPs:NUM?* to query the number of output groups and send *:TIMEr:CYCLEs* to set the number of cycles.

Return Format

The query returns OFF or LAST.

Examples

```
:TIME:ENDS LAST /*Sets the end state to "LAST".*/
:TIME:ENDS? /*Queries the end state. The query returns LAST.*/
```

4.12.4 :TIMEr:GROUPs:NUM?

Syntax

`:TIMEr:GROUPs:NUM?`

Description

Queries the number of output groups in the Arb editor.

Parameter

None.

Remarks

- The number of output groups refers to the number of preset voltage/current groups in each cycle.
- The total number of groups in timing output = the number of output groups × repetition cycle; wherein, you can send *:TIMEr:CYCLEs* to set the repetition cycle.
- The power supply terminates the timer function when the total number of groups is complete. At this point, the state of the power supply is decided by the setting in *:TIMEr:ENDState*.

Return Format

The query returns an integer between 1 and 512, for example, 25.

Examples

```
:TIME:GROUP:NUM? /*Queries the number of output groups. The query
returns 25.*/
```

4.12.5 TIMEr:GROUP:INDEx

Syntax

TIMEr:GROUP:INDEx <*val*>

TIMEr:GROUP:INDEx?

Description

Sets or queries the index number of the group currently edited in the Arb editor.

Parameter

Name	Type	Range	Default
<val>	Integer	1 to 512	-

Remarks

If there is no data inserted in the current row, it automatically goes to the last group of data in the editor.

Return Format

The query returns the index number of the group currently edited, for example, 25.

Examples

```
TIMEr:GROUP:INDEx 25 /*Sets the index number of the group currently
edited to 25./
TIMEr:GROUP:INDEx? /*The query returns 25.*/
```

4.12.6 :TIMEr:GROUP:PARAmeter

Syntax

:TIMEr:GROUP:PARAmeter <*volt*>,<*curr*>,<*time*>

:TIMEr:GROUP:PARAmeter? [<*groupcount*>]

Description

Inserts a group of data to the currently selected row in the Arb editor.

Queries the parameters of the group currently edited in the Arb editor.

Parameter

Name	Type	Range	Default
<volt>	Real	The voltage range of the current channel	-

Name	Type	Range	Default
<curr>	Real	The current range of the current channel	-
<time>	Real	Up to 3600 s	-
<groupcount>	Integer	1 to 512	1

Remarks

- <volt>, <curr>, and <time> are the voltage, current, and time of the group and their units are V, A, and s respectively.
- <groupcount> is the number of the group of parameters to be queried and the command queries from the index number set in *TIMEr:GROUP:INDEx*. The output stops with insufficient data.

Return Format

The query returns a string starting with **#**.

For example, the query might return #90000000431,0.500,1.0000,1.000;2,5.500,2.0000,1.000;; wherein, #9000000043 is the data block header, and 1,0.500,1.0000,1.000;2,5.500,2.0000,1.000; are the specified Arb parameters.

- In the format of #NX...X, the data block header is used to describe the length information. For example, in #9000000043, the N is 9, indicating that the 9 numbers following it are intended to describe the data length. That is, 000000043 can indicate the length of this data block (43 bytes).
- Each group of parameters is in the format of "number,voltage,current,time" , and multiple groups of parameters are separated by ";". For example, 1,0.500,1.0000,1.000;2,5.500,2.0000,1.000; are two groups of parameters. The number of the first group is 1, with 0.5 V voltage, 1 A current, and 1 s time; the number of the second group is 2, with 5.5 V voltage, 2 A current, and 1 s time.

Examples

```
:TIMEr:GROUP:INDEx 1 /*Sets the index number of the group currently
edited to 1.*/
:TIME:GROUP:PARA 0.5,1,1 /*Sets the parameters of the group
currently edited to 0.5 V, 1 A, and 1 s.*/
:TIMEr:GROUP:INDEx 2 /*Sets the index number of the group currently
edited to 2.*/
:TIME:GROUP:PARA 5.5,2,1 /*Sets the parameters of the group
currently edited to 5.5 V, 2 A, and 1 s.*/
:TIMEr:GROUP:INDEx 1 /*Sets the index number of the group currently
edited to 1.*/
:TIME:GROUP:PARA? 2 /*Queries two groups of parameters starting
```

```
from group 1. The query returns
#90000000431,0.500,1.0000,1.000;2,5.500,2.0000,1.000.*/
```

4.12.7 :TIMEr:GROUP:DELete

Syntax

:TIMEr:GROUP:DELete [<*groupcount*>]

Description

Deletes groups of parameters starting from the group currently edited.

Parameter

Name	Type	Range	Default
<groupcount>	Integer	1 to 512	1

Remarks

<Groupcount> is the number of groups to be deleted. If not specified, it is 1 by default.

Return Format

None.

Examples

```
:TIMEr:GROUP:INDEx 25 /*Sets the index number of the group
currently edited to 25./
:TIMEr:GROUP:DELete 2 /*Deletes the 25th and 26th group of
parameters.*/
```

4.12.8 :TIMEr:RUN

Syntax

:TIMEr:RUN <*run*>

:TIMEr:RUN?

Description

Sets or queries the run type of the generator.

Parameter

Name	Type	Range	Default
<run>	Discrete	{CONTinue|SINGle}	CONTinue

Remarks

- **CONTinue:** The instrument will output waveforms continuously according to the number of groups and repetition cycle currently set when the waveform output is enabled.
- **SINGle:** The instrument will output a single set of data in order each time the waveform output is enabled.

Return Format

The query returns CONTINUE or SINGLE.

Examples

```
:TIMEr:RUN SINGle /*Sets the run type to single.*/
:TIMEr:RUN? /*Queries the run type. The query returns SINGle.*/
```

4.12.9 :TIMEr[:STATe]

Syntax

:TIMEr[:STATe] <*bool*>

:TIMEr[:STATe]?

Description

Sets or queries the run/stop state of the generator.

Parameter

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

Remarks

- Turning on the generator will change the channel output state. Please make sure that the change in the output state will not affect the device connected to the power supply before enabling the output.
- The arbitrary waveform output is valid only when both the generator and the selected channel are turned on.
- When the run mode (*:TIMEr:RUN*) is set to "Continuous" (CONTinue), turning on the selected channel and the generator (:TIME ON), the instrument will repeat the sequence continuously based on the number of data groups and repetition cycle currently set. If the trigger source (*:TIMEr:TRIG*) is set to "BUS", you also need to send **TRG* to enable the output.
- When the run mode (*:TIMEr:RUN*) is set to "Single" (SINGle), turning on the selected channel and the generator (:TIME ON), the instrument will output a single group of data in order each time :TIME ON is sent. If the trigger source

(*:TIMEr:TRIG*) is set to "BUS", the instrument will output a single group of data each time **TRG* is sent.

- While the generator is turned on, Arb parameters cannot be modified.

Return Format

The query returns 1 or 0.

Examples

```
:TIME ON /*Turns on the generator.*/
:TIME? /*Queries the run/stop state of the generator. The query
returns 1.*/
```

4.12.10 :TIMEr:TEMPlet:CONSTruct

Syntax

`:TIMEr:TEMPlet:CONSTruct`

Description

Sets the Arb parameters based on the template currently selected and the parameters set.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
:TIMEr:TEMPlet:CONSTruct /*Sets the Arb parameters based on the
template currently selected and the parameters set.*/
```

4.12.11 :TIMEr:TEMPlet:FALLRate

Syntax

`:TIMEr:TEMPlet:FALLRate` <*value*>

`:TIMEr:TEMPlet:FALLRate?`

Description

Sets or queries the fall index of ExpFall.

Parameter

Name	Type	Range	Default
<value>	Integer	0 to 10	0

Remarks

When the template currently selected is ExpFall (*:TIMEr:TEMPlet:SELect*), the parameters set cannot reach the minimum due to the characteristic of the exponential function itself. The range of the parameters created is related to the fall index currently set. The larger the fall index is, the wider the range of the parameters will be.

Return Format

The query returns an integer between 0 and 10, for example, 5.

Examples

```
:TIME:TEMP:FALLR 5 /*Sets the fall index of ExpFall to 5.*/
:TIME:TEMP:FALLR? /*Queries the fall index of ExpFall currently
set. The query returns 5.*/
```

4.12.12 :TIMEr:TEMPlet:INTErval

Syntax

:TIMEr:TEMPlet:INTErval <*value*>

:TIMEr:TEMPlet:INTErval?

Description

Sets or queries the Time Interval.

Parameter

Name	Type	Range	Default
<value>	Real	0.001 s to 3600 s	1 s

Remarks

- Time interval refers to the time required for the instrument to output each group of parameters using the template currently selected.
- This command is not available for Stair Up, Stair Dn, Stair UpDn, and Pulse.

Return Format

The query returns a real number, for example, 15.000.

Examples

```
:TIME:TEMP:INTE 15 /*Sets the Time Interval to 15 s.*/
:TIME:TEMP:INTE? /*Queries the Time Interval currently set. The
query returns 15.000.*/
```

4.12.13 :TIMEr:TEMPlet:INVErt

Syntax

:TIMEr:TEMPlet:INVErt <*bool*>

:TIMEr:TEMPlet:INVErt?

Description

Sets or queries the on/off state of the invert function of the template currently selected.

Parameter

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

Remarks

- When the invert function is enabled, the instrument will first invert the waveform and then set the waveform parameters.
- The invert function is available for Sine, Pulse, and Ramp templates only.

Return Format

The query returns 1 or 0.

Examples

```
:TIME:TEMP:INVE ON /*Enables the invert function for the template
currently selected.*/
:TIME:TEMP:INVE? /*Queries whether the invert function is enabled.
The query returns 1.*/
```

4.12.14 :TIMEr:TEMPlet:MAXValue

Syntax

:TIMEr:TEMPlet:MAXValue {<*value*>|*MINimum*|*MAXimum*}

:TIMEr:TEMPlet:MAXValue? [*MINimum*|*MAXimum*]

Description

Sets or queries the maximum voltage or current value of the selected template.

Parameter

Name	Type	Range	Default
<value>	Real	Voltage or current range of the channel currently selected	1V/1A

Remarks

- When the object (*:TIMEr:TEMPlet:OBJect*) is set to Voltage (V), this command sets the maximum voltage value; when the object is set to Current (C), this command sets the maximum current value.
- When the selected template is Pulse, this command sets or queries the high level.
- "MINimum" and "MAXimum" indicate the minimum and maximum current/voltage available.

Return Format

The query returns the maximum voltage or current of the template currently selected, for example, 5.000 or 5.3000.

Examples

```
:TIME:TEMP:OBJ V,2 /*Sets the object to voltage and the current
value to 2 A.*/
:TIME:TEMP:MAXV 5 /*Sets the maximum voltage to 5 V for the
selected template.*/
:TIME:TEMP:MAXV? /*Queries the maximum voltage value for the
template currently selected. The query returns 5.000.*/
```

4.12.15 :TIMEr:TEMPlet:MINValue

Syntax

:TIMEr:TEMPlet:MINValue {<*value*>|*MINimum*|*MAXimum*}

:TIMEr:TEMPlet:MINValue? [*MINimum*|*MAXimum*]

Description

Sets or queries the minimum voltage or current value of the template currently selected.

Parameter

Name	Type	Range	Default
<value>	Real	Voltage or current range of the channel currently selected	0

Remarks

- When the object (*:TIMEr:TEMPlet:OBJect*) is set to Voltage (V), this command sets the minimum voltage value; when the object is set to Current (C), this command sets the minimum current value.
- When the selected template is Pulse, this command sets or queries the low level.
- "MINimum" and "MAXimum" indicate the minimum and maximum current/voltage available.

Return Format

The query returns the minimum voltage or current value of the template currently selected, for example, 0.200 or 0.1000.

Examples

```
:TIME:TEMP:OBJ C,1.5 /*Sets the object to current and the voltage
value to 1.5 V.*/
:TIME:TEMP:MINV 0.1 /*Sets the minimum current to 0.1 A for the
selected template.*/
:TIME:TEMP:MINV? /*Queries the minimum current for the template
currently selected. The query returns 0.1000.*/
```

4.12.16 :TIMEr:TEMPlet:OBJect

Syntax

:TIMEr:TEMPlet:OBJect <*obj*>[,<*value*>|*MINimum*|*MAXimum*]

:TIMEr:TEMPlet:OBJect? [*MINimum*|*MAXimum*]

Description

Sets or queries the editing object of the template currently selected as well as the corresponding current or voltage value.

Parameter

Name	Type	Range	Default
<obj>	Discrete	{V|C}	V
<value>	Real	Voltage or current range of the channel currently selected	0

Remarks

- Setting <obj> to "V" indicates selecting voltage to edit. <value> is used to set the constant current value and its range is the current range of the selected channel. At this point, you can send *:TIMEr:TEMPlet:MAXValue*

and *:TIMEr:TEMPlet:MINValue* to set the maximum and minimum values of voltage respectively.

- Setting <obj> to "C" indicates selecting current to edit. <value> is used to set the constant voltage value and its range is the voltage range of the selected channel. At this point, you can send *:TIMEr:TEMPlet:MAXValue* and *:TIMEr:TEMPlet:MINValue* to set the maximum and minimum values of current respectively.

Return Format

The query returns the editing object currently selected and the corresponding current or voltage value (separated by comma). For example, the query might return V,2.0000. Wherein, "V" indicates selecting voltage to edit, and 2.0000 indicates setting the constant current value to 2 A.

Examples

```
:TIME:TEMP:OBJ V,2 /*Selects the editing object to voltage and sets
the constant current to 2 A.*/
:TIME:TEMP:OBJ? /*Queries the editing object and the corresponding
constant current or voltage value. The query returns V,2.0000.*/
```

4.12.17 :TIMEr:TEMPlet:PERIod

Syntax

:TIMEr:TEMPlet:PERIod <*value*>

:TIMEr:TEMPlet:PERIod?

Description

Sets or queries the Period of the waveform.

Parameter

Name	Type	Range	Default
<value>	Real	0.001 s to 3600 s	Refer to *Remarks*

Remarks

- <value> specifies the duration of a cycle for a waveform. All waveforms except for ExpRise and ExpFall require a period.
- By default, the period of Pulse is 2 s, and the period of other waveforms is 50 s.
- The total number of points within a period of the selected template is determined by the period of the waveform and time interval

(*:TIMEr:TEMPlet:INTErval*). The number of points within a period=period/time interval.

Return Format

The query returns a real number ranging from 0.001 to 3600, for example, 15.000.

Examples

```
:TIME:TEMP:PERI 15 /*Sets the Period to 15 s.*/
:TIME:TEMP:PERI? /*Queries the Period for the selected template.
The query returns 15.000.*/
```

4.12.18 :TIMEr:TEMPlet:POINTs

Syntax

:TIMEr:TEMPlet:POINTs <*value*>

:TIMEr:TEMPlet:POINTs?

Description

Sets or queries the total number of points.

Parameter

Name	Type	Range	Default
<value>	Integer	1 to 512	50

Remarks

- The total number of points refers to the number of groups of parameters created based on the template currently selected.
- When the total number of points (denoted by **P**) and the number of current output groups (denoted by **G**, *:TIMEr:GROUPs:NUM?*) are different, **P** groups of parameters will be created using the template; then, the number of output groups will change to **P** automatically.

Return Format

The query returns an integer between 1 and 512, for example, 80.

Examples

```
:TIME:TEMP:POINT 80 /*Sets the total number of points to 80.*/
:TIME:TEMP:POINT? /*Queries the total number of points currently
set. The query returns 80.*/
```

4.12.19 :TIMEr:TEMPlet:RISERate

Syntax

:TIMEr:TEMPlet:RISERate <*value*>

:TIMEr:TEMPlet:RISERate?

Description

Sets or queries the rise index of ExpRise.

Parameter

Name	Type	Range	Default
<value>	Integer	0 to 10	0

Remarks

When the template currently selected is ExpRise (*:TIMEr:TEMPlet:SELect*), the parameters set cannot reach the maximum due to the characteristic of the exponential function itself. The range of the parameters created is related to the rise index currently set. The larger the rise index is, the wider the range of the parameters will be.

Return Format

The query returns an integer between 0 and 10, for example, 5.

Examples

```
:TIME:TEMP:RISER 5 /*Sets the rise index of ExpRise to 5.*/
:TIME:TEMP:RISER? /*Queries the rise index of ExpRise currently
set. The query returns 5.*/
```

4.12.20 :TIMEr:TEMPlet:SELect

Syntax

:TIMEr:TEMPlet:SELect <*temp*>

:TIMEr:TEMPlet:SELect?

Description

Sets or queries template type.

Parameter

Name	Type	Range	Default
<temp>	Discrete	{SINE|PULSE|RAMP|UP|DN|UPDN|RISE|FALL}	SINE

Remarks

None.

Return Format

The query returns SINE, PULSE, RAMP, UP, DN, UPDN, RISE, or FALL.

Examples

```
:TIME:TEMP:SEL UP /*Selects the Stair Up template.*/
:TIME:TEMP:SEL? /*Queries the template currently selected. The
query returns UP.*/
```

4.12.21 :TIMEr:TEMPlet:SYMMetry

Syntax

:TIMEr:TEMPlet:SYMMetry <*value*>

:TIMEr:TEMPlet:SYMMetry?

Description

Sets or queries the Symmetry of Ramp.

Parameter

Name	Type	Range	Default
<value>	Integer	0 to 100	50

Remarks

Symmetry is specified as the ratio of the duration of the rising edge within a period to the whole period.

Return Format

The query returns the symmetry, for example, 60%.

Examples

```
:TIME:TEMP:SYMM 60 /*Sets the Symmetry to 60%.*/
:TIME:TEMP:SYMM? /*Queries the Symmetry. The query returns 60%.*/
```

4.12.22 :TIMEr:TEMPlet:WIDTh

Syntax

:TIMEr:TEMPlet:WIDTh <*value*>

:TIMEr:TEMPlet:WIDTh?

Description

Sets or queries the Positive Pulse Width of Pulse.

Parameter

Name	Type	Range	Default
<value>	Real	Up to 3600 s	1 s

Remarks

- Pulse width refers to the duration of high level within a period.
- The actual available range of <value> is related to the period currently set (*:TIMEr:TEMPlet:PERIod*). The positive pulse width cannot be larger than the period.

Return Format

The query returns a real number, for example, 14.000.

Examples

```
:TIME:TEMP:WIDT 14 /*Sets the Pulse Width to 14 s.*/
:TIME:TEMP:WIDT? /*Queries the Pulse Width. The query returns
14.000.*/
```

4.12.23 :TIMEr:TEMPlet:STAIr

Syntax

:TIMEr:TEMPlet:STAIr <*val*>

:TIMEr:TEMPlet:STAIr?

Description

Sets or queries the number of steps between the maximum and minimum for StairUp, StairDn, or StairUpDn.

Parameter

Name	Type	Range	Default
<val>	Integer	3 to 99999	25

Remarks

To create a StairUp, StairDn, or StairUpDn waveform with a complete cycle, <val> should be smaller than the number of the group of parameters (*:TIMEr:TEMPlet:POINTs*).

Return Format

The query returns the number of steps between the maximum and minimum for the waveform, for example, 30.

Examples

```
:TIME:TEMP:SEL UP /*Selects the StairUp template.*/
:TIMEr:TEMPlet:STAIr 30 /*Sets the number of steps between the
maximum and minimum for StairUp to 30.*/
:TIMEr:TEMPlet:STAIr? /*Queries the number of steps between the
maximum and minimum for StairUp. The query returns 30.*/
```

4.12.24 :TIMEr:TRIG

Syntax

:TIMEr:TRIG <*trig*>

:TIMEr:TRIG?

Description

Sets or queries the Trigger Source.

Parameter

Name	Type	Range	Default
<trig>	Discrete	{MANual|BUS}	MANual

Remarks

Trigger source specifies the way of starting the output of the arbitrary waveform. Options include "Manual" and "BUS".

- **MANual** selects the Run/Stop key as a trigger source. When both the selected channel (*:OUTPut[:STATe]*) and generator (*:TIMEr[:STATe]*) is turned on, the instrument will output waveforms based on the selected run mode (*:TIMEr:RUN*).
- **BUS** selects a remote command as a trigger source. The instrument waits for the trigger signal after the generator (*:TIMEr[:STATe]*) is turned on. After the selected channel is turned on, the instrument will output waveforms based on the set run mode (*:TIMEr:RUN*) when **TRG* is received.

Return Format

The query returns MANUAL or BUS.

Examples

```
:TIMEr:TRIG BUS /*Sets the Trigger Source to BUS.*/
:TIMEr:TRIG? /*Queries the Trigger Source. The query returns BUS.*/
```

4.13 :TRIGger Commands

:TRIGger commands are used to enable and disable the trigger, set and query trigger conditions as well as the source under control and control source of the specified data line.

4.13.1 :TRIGger:IN[:ENABle]

Syntax

:TRIGger:IN[:ENABle] <*d*>,<*bool*>

:TRIGger:IN[:ENABle]? <*d*>

Description

Sets or queries the on/off state of the trigger input function for the specified data line.

Parameter

Name	Type	Range	Default
<d>	Discrete	{D1|D2|D3|D4}	D1
<bool>	Bool	{1|ON|0|OFF}	0|OFF

Remarks

When the specified data line receives input signal that meets the current trigger type (*:TRIGger:IN:TYPE*), the specified source under control (*:TRIGger:IN:SOURce*) will turn on/off the output, or toggle the output state according to the setting in *:TRIGger:IN:RESPonse*.

Return Format

The query returns 1 or 0.

Examples

```
:TRIG:IN D1,ON /*Enables the trigger input function for D1.*/
:TRIG:IN? D1 /*Queries the on/off state of the trigger input
function for D1. The query returns 1.*/
```

4.13.2 :TRIGger:IN:IMMEdiate

Syntax

`:TRIGger:IN:IMMEdiate`

Description

As soon as the trigger system is initiated, the analog hardware will send the trigger signal immediately.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

None.

4.13.3 :TRIGger:IN:RESPonse

Syntax

`:TRIGger:IN:RESPonse` <*d*>,<*res*>

`:TRIGger:IN:RESPonse?` <*d*>

Description

Sets or queries the output response of the trigger input for the specified data line.

Parameter

Name	Type	Range	Default
<d>	Discrete	{D1|D2|D3|D4}	D1
<res>	Discrete	{ON|OFF|ALTER}	OFF

Remarks

- **OutOn (ON):** Turns on the output of the channel currently selected as the source under control (*:TRIGger:IN[:ENABle]*) when the trigger condition (*:TRIGger:IN:TYPE*) is met.

- **OutOff (OFF):** Turns off the output of the channel currently selected as the source under control when the trigger condition (*:TRIGger:IN:TYPE*) is met.
- **OutFlip (ALTER):** Toggles the channel output state when the trigger condition (*:TRIGger:IN:TYPE*) is met. That is, turns off the channel when the current channel is on, or turns on the channel when the current channel is off.

Return Format

The query returns ON, OFF, or ALTER.

Examples

```
:TRIG:IN:RESP D1,ON /*Sets the output response of D1 trigger input
to OutOn.*/
:TRIG:IN:RESP? D1 /*Queries the output response of D1 trigger
input. The query returns ON.*/
```

4.13.4 :TRIGger:IN:SOURce

Syntax

:TRIGger:IN:SOURce <*d*>,<*ch*>

:TRIGger:IN:SOURce? <*d*>

Description

Sets or queries the source under control of the trigger input for the specified data line.

Parameter

Name	Type	Range	Default
<d>	Discrete	{D1|D2|D3|D4}	D1
<ch>	ASCII string	{CH1[,CH2[,CH3]]}	CH1

Remarks

You can select one or more channels from CH1, CH2, and CH3 as the source under control. If <ch> is omitted, CH1 is selected as the source under control.

Return Format

The query returns the name of the source under control. If the source under control contains multiple channels, the channels are separated by commas. For example, the query might return CH1 or CH1,CH2.

Examples

```
:TRIG:IN:SOUR D1,CH1,CH2 /*Sets the source under control of D1
trigger input to CH1 and CH2.*/
```

```
:TRIG:IN:SOUR? D1 /*Queries the source under control of D1 trigger
input. The query returns CH1,CH2.*/
```

4.13.5 :TRIGger:IN:TYPE

Syntax

:TRIGger:IN:TYPE <*d*>,<*type*>

:TRIGger:IN:TYPE? <*d*>

Description

Sets or queries the trigger type of the trigger input for the specified data line.

Parameter

Name	Type	Range	Default
<d>	Discrete	{D1|D2|D3|D4}	D1
<type>	Discrete	{RISE|FALL|HIGH|LOW}	RISE

Remarks

- You can select to trigger on the rising edge (RISE), falling edge (FALL), high level (HIGH), or low level (LOW) of the input signal.
- For the input signal, high level ranges from 2.5 V to 3.3 V, low level from 0 V to 0.8 V, and the noise tolerance is 0.4 V.

Return Format

The query returns RISE, FALL, HIGH, or LOW.

Examples

```
:TRIG:IN:TYPE D1,FALL /*Sets the trigger type of D1 trigger input
to the falling edge.*/
:TRIG:IN:TYPE? D1 /*Queries the trigger type of D1 trigger input.
The query returns FALL.*/
```

4.13.6 :TRIGger:OUT:POLArity

Syntax

:TRIGger:OUT:POLArity <*d*>,<*pol*>

:TRIGger:OUT:POLArity? <*d*>

Description

Sets or queries the output response of the trigger output for the specified data line.

Parameter

Name	Type	Range	Default
<d>	Discrete	{D1|D2|D3|D4}	D1
<pol>	Discrete	{POSitive|NEGative}	POSitive

Remarks

- **POSitive:** The selected data line outputs 3.3 V high level signal when the control source is turned on.
- **NEGative:** The selected data line outputs low level signal (CMOS level) when the control source is turned on.

Return Format

The query returns POSITIVE or NEGATIVE.

Examples

```
:TRIG:OUT:POLA D1,NEGAtive /*Sets the D1 trigger output signal to
low level.
:TRIG:OUT:POLA? D1 /*Queries the D1 trigger output signal. The
query returns NEGATIVE.*/
```

4.13.7 :TRIGger:OUT:SOURce

Syntax

:TRIGger:OUT:SOURce <*d*>,<*source*>

:TRIGger:OUT:SOURce? <*d*>

Description

Sets or queries the control source of the trigger output function of the specified data line.

Parameter

Name	Type	Range	Default
<d>	Discrete	{D1|D2|D3|D4}	D1
<source>	Discrete	{CH1|CH2|CH3}	CH1

Remarks

You can select any one of CH1, CH2, and CH3 as the control source of trigger output.

Return Format

The query returns the name of the control source selected, for example, CH1.

Examples

```
:TRIG:OUT:SOUR D1,CH1 /*Sets the control source of D1 trigger
output to CH1.*/
:TRIG:OUT:SOUR? D1 /*Queries the control source of D1 trigger
output. The query returns CH1.*/
```

4.13.8 :TRIGger:OUT[:ENABle]

Syntax

:TRIGger:OUT[:ENABle] <*d*>,<*bool*>

:TRIGger:OUT[:ENABle]? <*d*>

Description

Sets or queries the on/off state of the trigger output function for the specified data line.

Parameter

Name	Type	Range	Default
<d>	Discrete	{D1|D2|D3|D4}	D1
<bool>	Bool	{0|1|ON|OFF}	0|OFF

Remarks

After the trigger output function is enabled, the specified data line outputs high/low level signal according to the settings in *:TRIGger:OUT:POLArity* when the specified control source (*:TRIGger:OUT:SOURce*) is turned on.

Return Format

The query returns 0 or 1.

Examples

```
:TRIG:OUT D1,ON /*Enables the trigger output function of D1.*/
:TRIG:OUT? D1 /*Queries the on/off state of the D1 trigger output
function. The query returns 1.*/
```

5 Programming Examples

This chapter illustrates how to control the instrument by programming in LabVIEW, Visual Basic, and Visual C++. These examples are programmed based on Virtual Instrument Software Architecture (VISA) library.

5.1 Programming Preparations

Before programming, you need to prepare the following tasks:

You can log in to the RIGOL official website (*http://www.rigol.com*) to download the software. Then install the software according to the installation wizard. After Ultra Sigma is installed successfully, NI-VISA library will be completely installed automatically. In this manual, the default installation path is C:\Program Files\IVI Foundation\VISA.

In the manual, the instrument communicates with the PC via the USB interface. Connect the USB Device interface on the rear panel of the instrument to the PC by using the USB cable. After the instrument is properly connected to the PC, power on the instrument to start it.

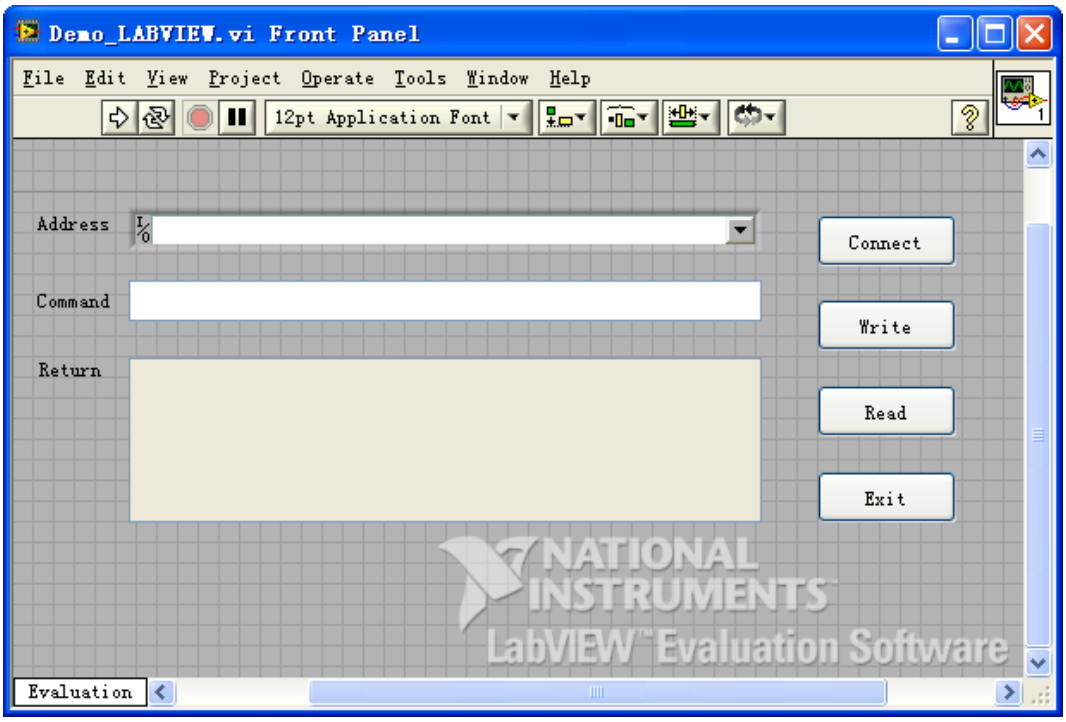
The following parts will make a detailed introduction about the programming examples in LabVIEW, Visual Basic, and Visual C++.

5.2 LabVIEW Programming Example

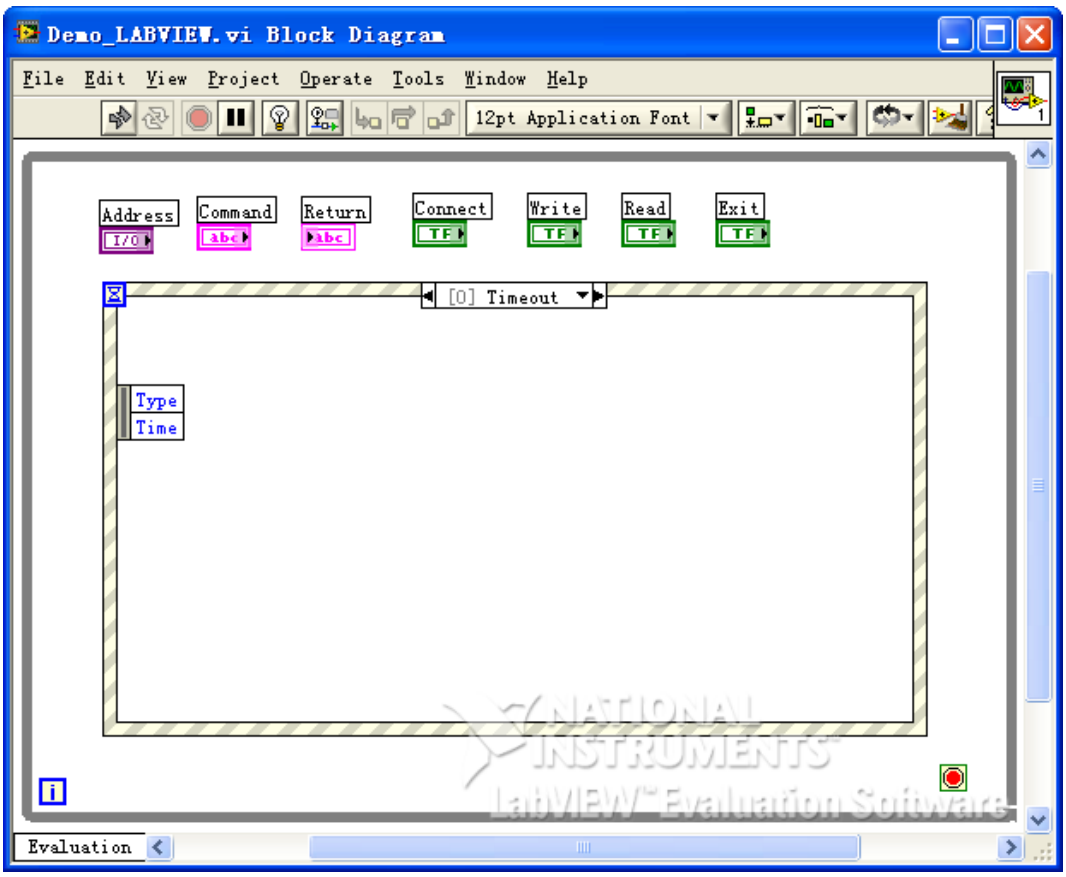
Program used in this example: LabVIEW 2009

Function realized in this example: search for the instrument address, connect the instrument, send command, and read the returned value.

1. Run LabVIEW, and then create a VI file named Demo_LABVIEW.
2. Add controls in the front panel interface, including the **Address**, **Command**, and **Return** field as well as the **Connect**, **Write**, **Read**, and **Exit** buttons.

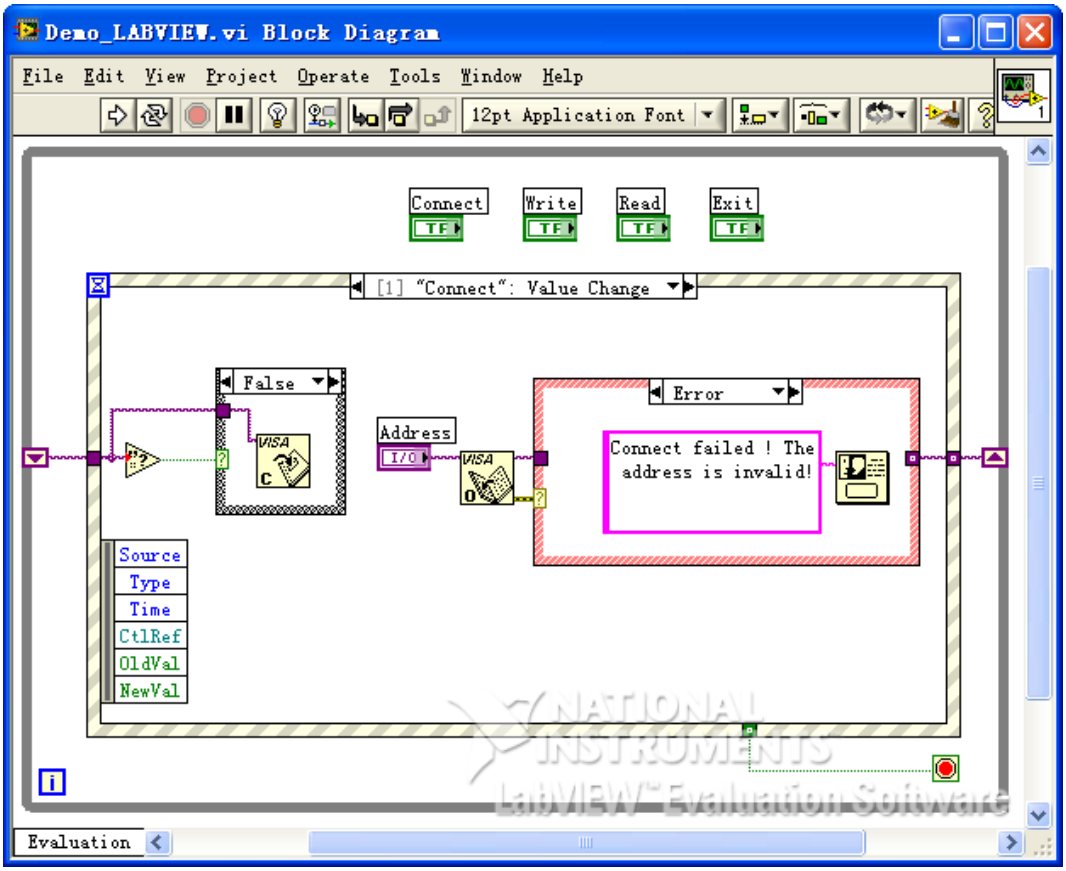


3. Click **Show Block Diagram** in the **Window** menu to create event structure.

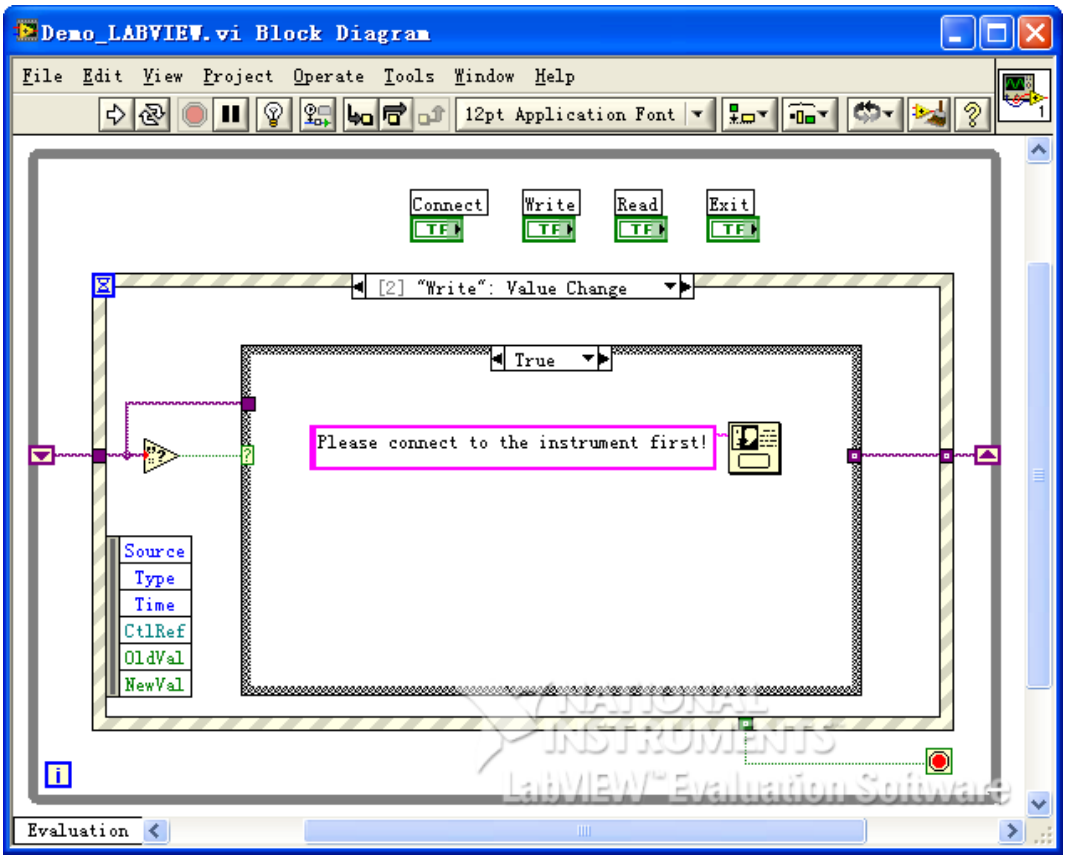


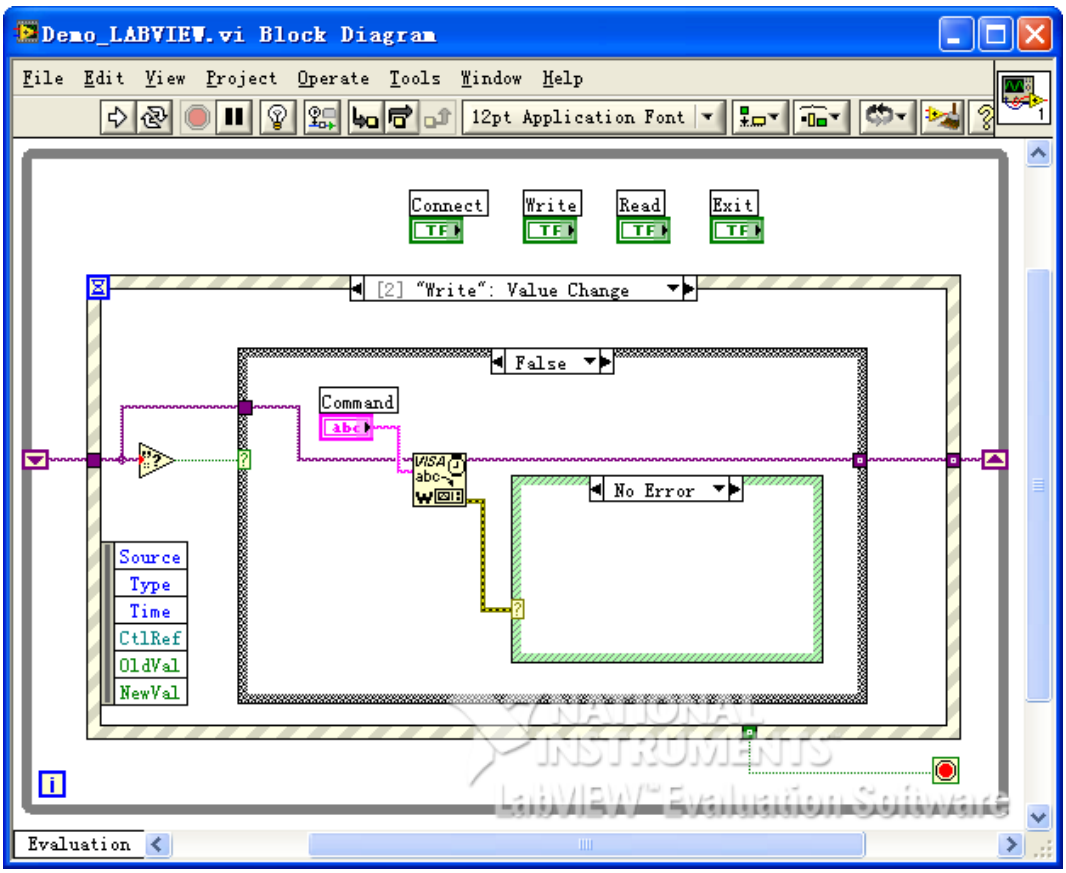
4. Add events, including connecting instrument, write operation, read operation, and exit.

a. Connect the instrument (including error processing):

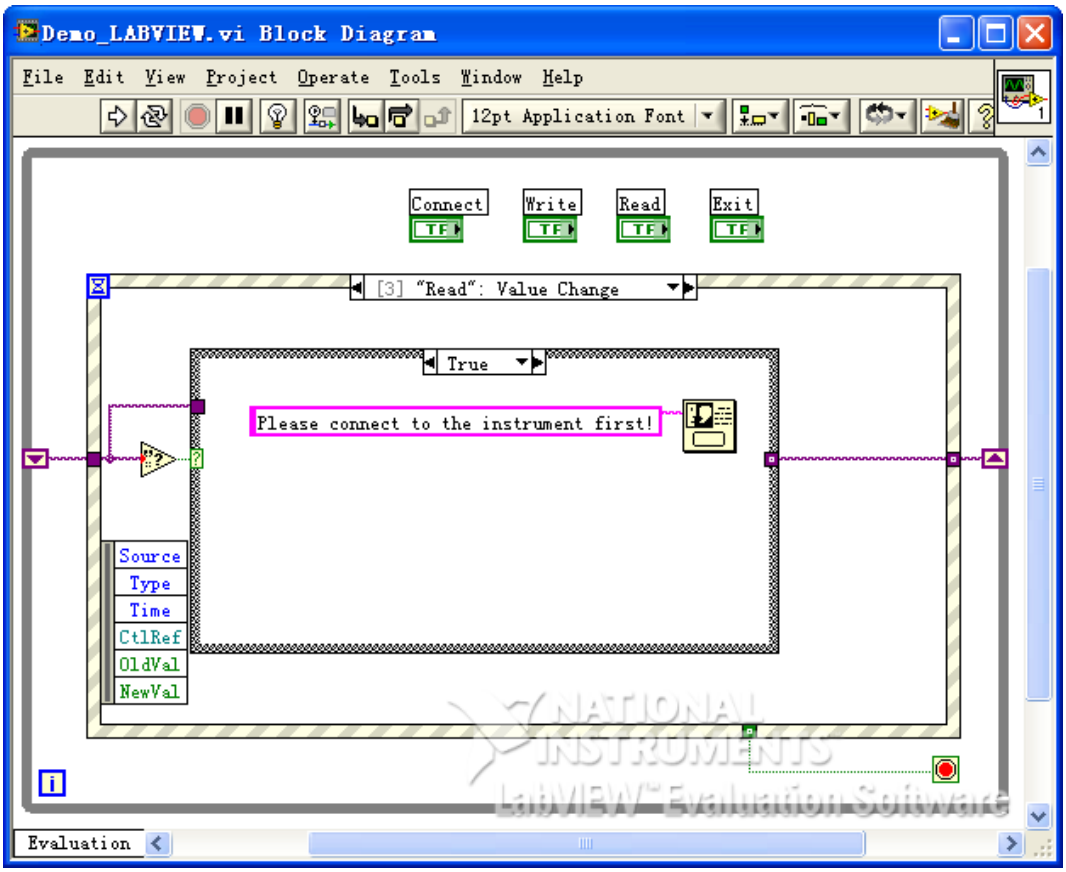


b. Write operation (including error judgment):

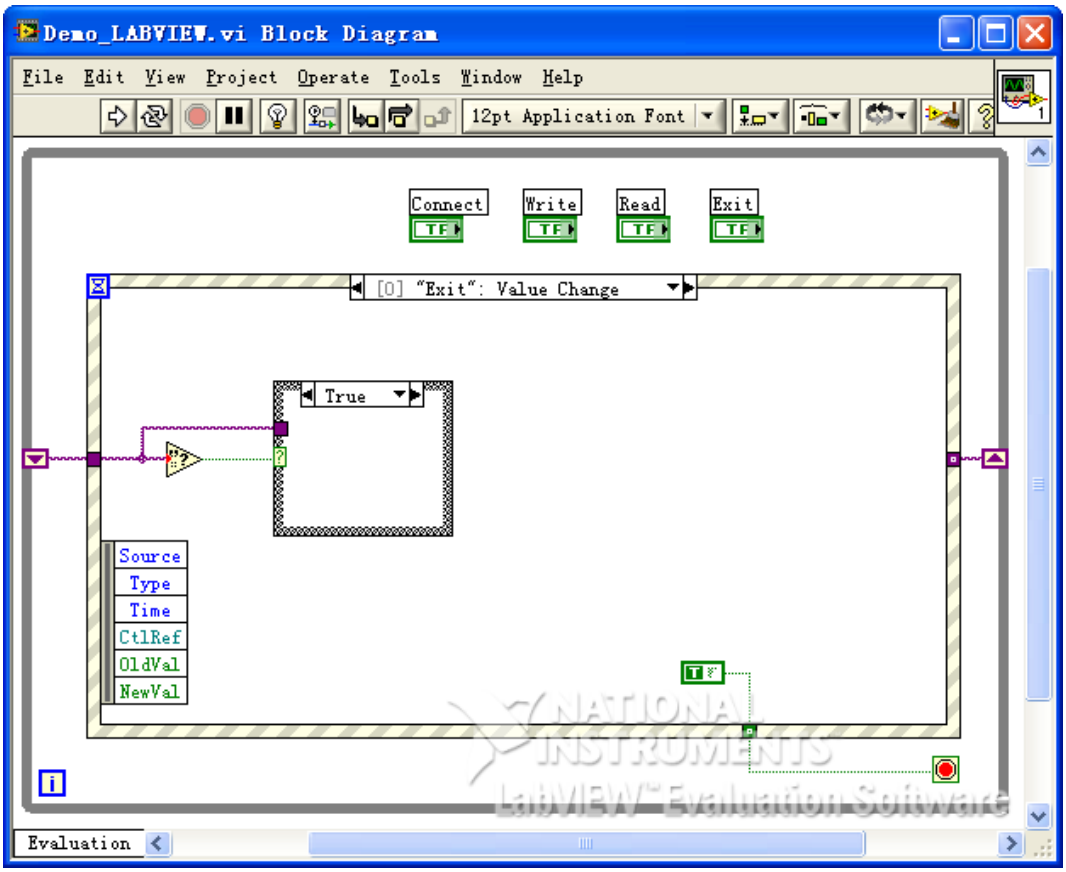




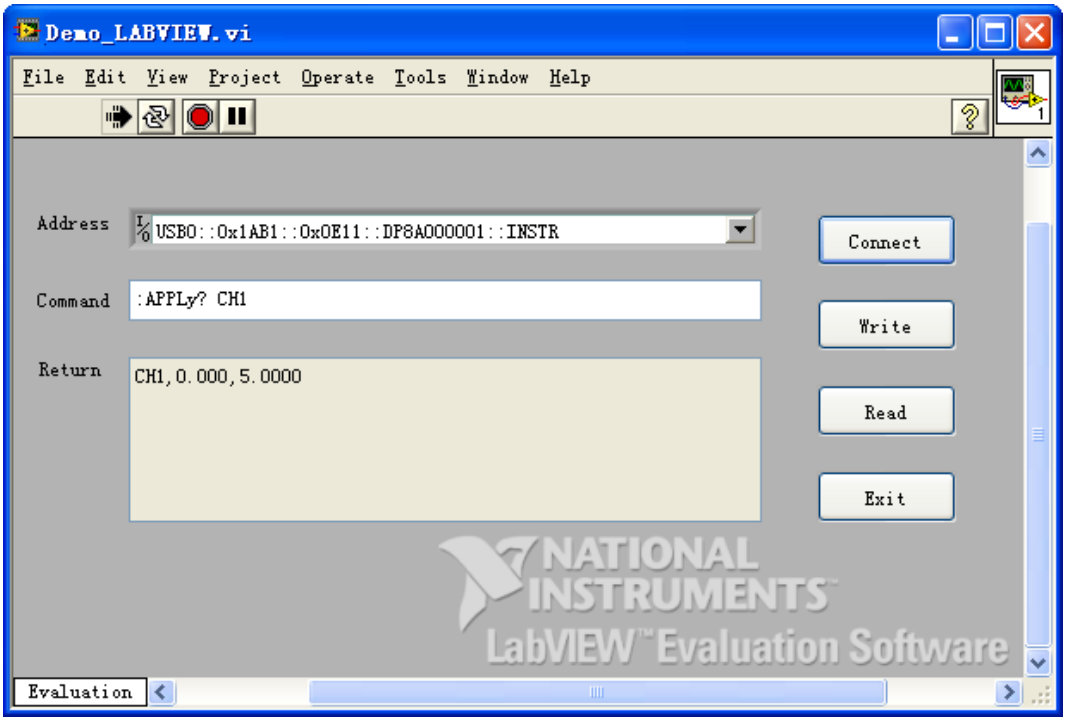
c. Read operation (including error processing):



d. Exit:



5. Run the program and the interface as shown in the figure below is displayed. Click the **Address** drop-down button and select the VISA resource name; click Connect to **connect** the instrument; enter the command into the **Command** input field and click **Write** to write the command into the instrument. If the command is a query command, click **Read** and the returned value is displayed in the **Return** field.



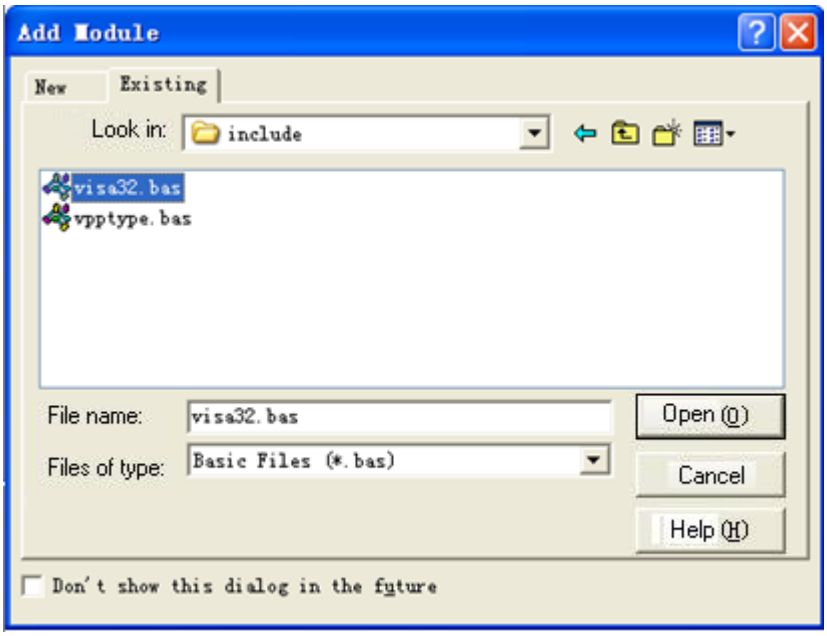
5.3 Visual Basic Programming Example

Program used in this example: Visual Basic 6.0

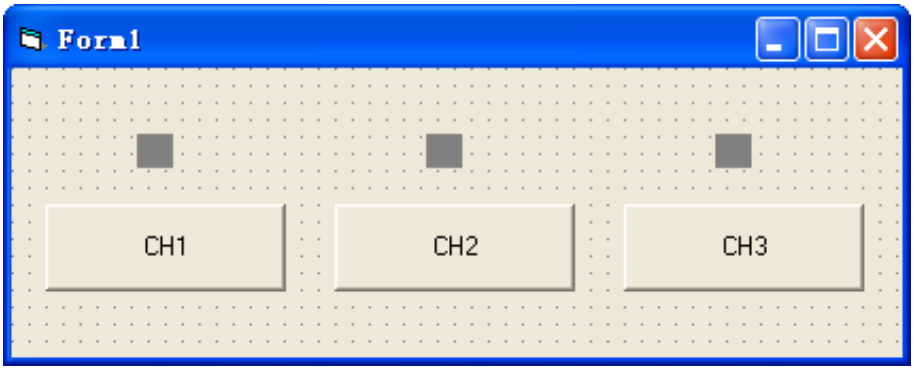
Function realized in this example: turn on the power supply’ s three channels with their color displayed.

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

1. Build a standard application program project (Standard EXE), and name it "Demo".
2. Click **Project** > **Add Module** to open the Add Module dialog box. In the dialog box, click the **Existing** tab to search for the **visa32.bas** file in the **include** folder under the **NI-VISA** installation path and add the file.



3. Add three **CommandButton** controls to represent **CH1**, **CH2** and **CH3** respectively. Add three **Text** controls (**Label1(0)**, **Label1(1)**, and **Label1(2)**) to represent the status of the three channels respectively (it displays gray by default; when the channel is enabled, it displays the color of the channel), as shown in the figure below.



4. Click **Project** > **Project1 Properties** to open the Project1 – Project Properties dialog box. In the dialog box, click on the **General** tab and select **Form1** from the drop-down button under **Startup Object**.

5. Double-click **CH1** to enter the programming environment. Add the following codes to control CH1-CH3. The codes of CH1 are as shown below; the codes of the other channels are similar.

```
Dim defrm As Long
Dim vi As Long
Dim strRes As String * 200
Dim list As Long
Dim nmatches As Long
Dim matches As String * 200 'Reserve the obtained device number
Dim s32Disp As Integer
' Obtain the usb resource of visa
Call viOpenDefaultRM(defrm)
```

```
Call viFindRsrc(defrm, "USB?*", list, nmatches, matches)
' Turn on the instrument
Call viOpen(defrm, matches, 0, 0, vi)
' Send a command to query the status of CH1
Call viVPrintf(vi, ":CHAN1:DISP?" + Chr$(10), 0)
' Obtain the status of CH1
Call viVScanf(vi, "%t", strRes)
s32Disp = CInt(strRes)
If (s32Disp = 1) Then
' Send the setting command
Call viVPrintf(vi, ":CHAN1:DISP 0" + Chr$(10), 0)
Label1(0).ForeColor = &H808080 ''Gray
Else
Call viVPrintf(vi, ":CHAN1:DISP 1" + Chr$(10), 0)
Label1(0).ForeColor = &HFFFF& 'Yellow
End If
' Close the resource
Call viClose(vi)
Call viClose(defrm)
```

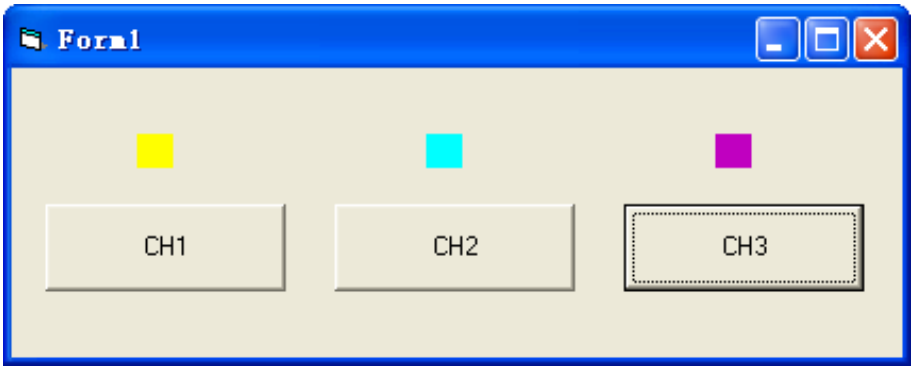
6. Results

   a. Click **CH1** to turn on CH1 and the label above **CH1** turns yellow;

   b. Click **CH2** to turn on CH2 and the label above **CH2** turns blue;

   c. Click **CH3** to turn on CH3 and the label above **CH3** turns rosy.

   The results are as shown in the figure below.



5.4 Visual C++ Programming Example

Program used in this example: Visual C++6.0

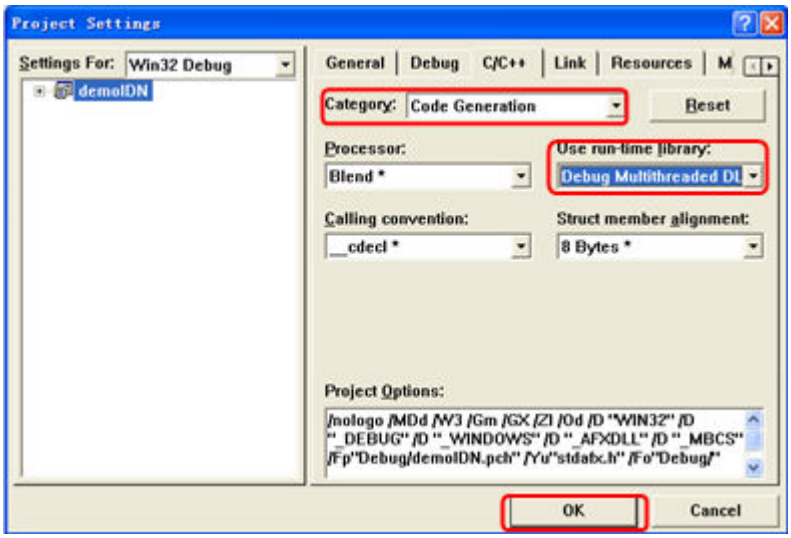
Function realized in this example:search for the instrument address, connect to the instrument, send commands, and read return values.

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

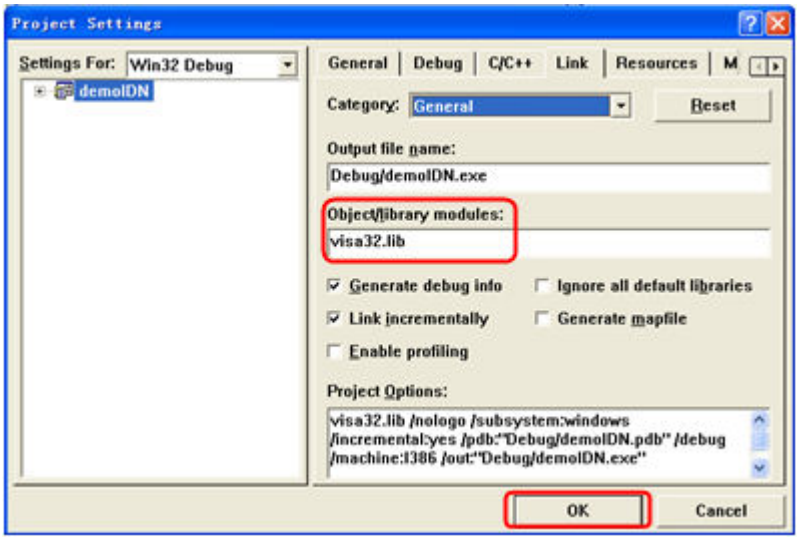
1. Create a MFC project based on a dialog box.

2. Click **Project** > **Settings** to open the **Project Setting** dialog box. In the dialog box, click the **C/C++** tab, select **Code Generation** from the drop-down list under

Category. Choose **Debug Multithreaded DLL** from the drop-down list under **Use run-time library**. Click **OK** to close the dialog box.



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

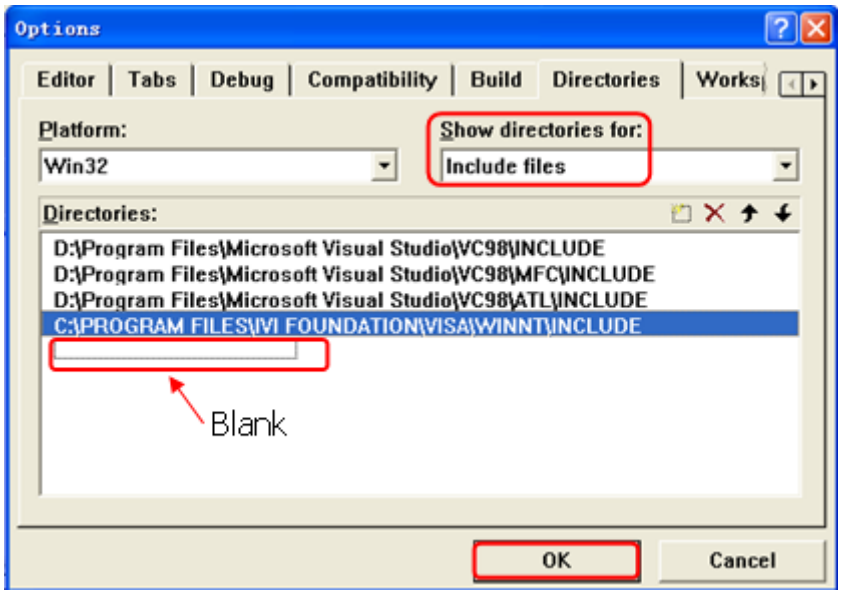


4. Click **Tools** > **Options** to open the Options dialog box. Then click the **Directories** tab.

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

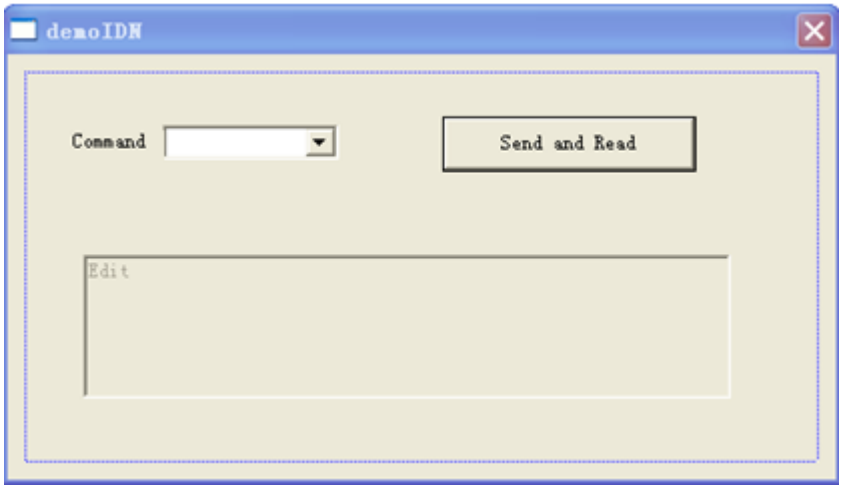
   Select **Library files** from the drop-down list under **Show directories for**. Double click the empty space under **Directories** to enter the specified path of Library files:

C:\Program Files\IVI Foundation\VISA\WinNT\lib\msc. Click **OK** to close the dialog box.



Note: By now, VISA library has been added.

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



6. Modify the control attributes.

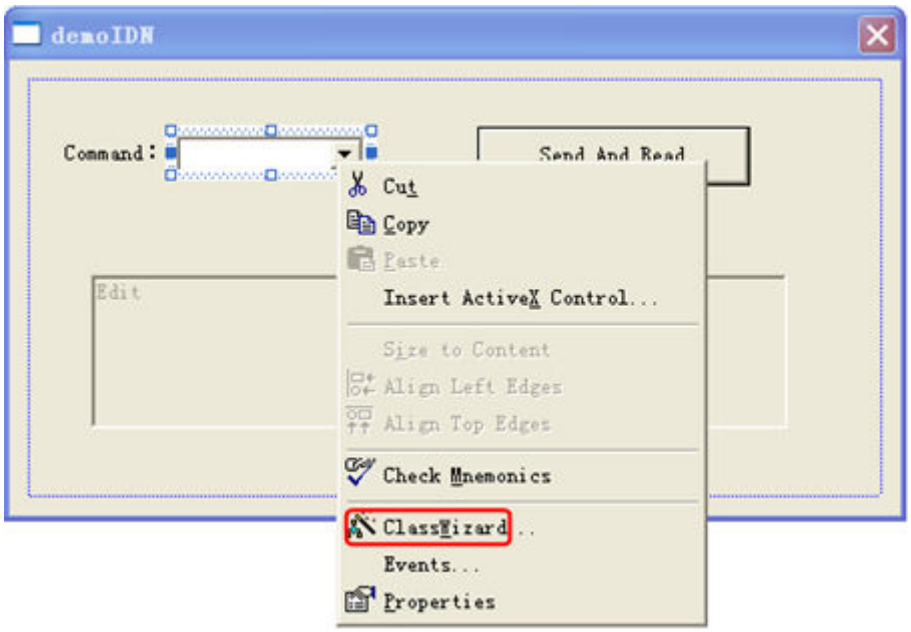
   a. Name **Text** as "Command".

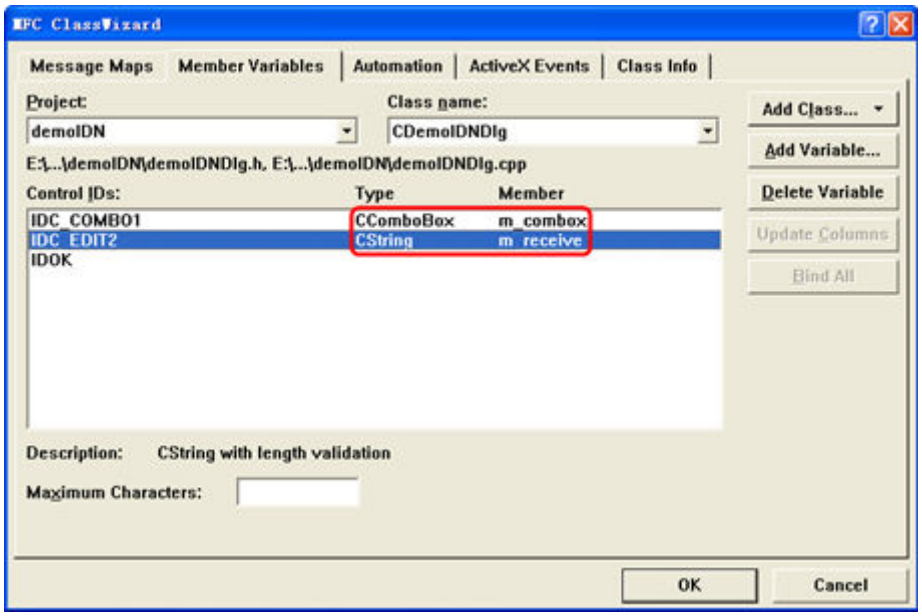
   b. Open the **Data** item in the **Combo Box** attribute and input the following command *IDN? manually.

   c. Open the **General** item in the **Edit Box** attribute and select **Disabled**.

   d. Name **Button** as **Send and Read**.

7. Add the variables m_combox and m_receive to the **Com Box** and **Edit Box** controls respectively.





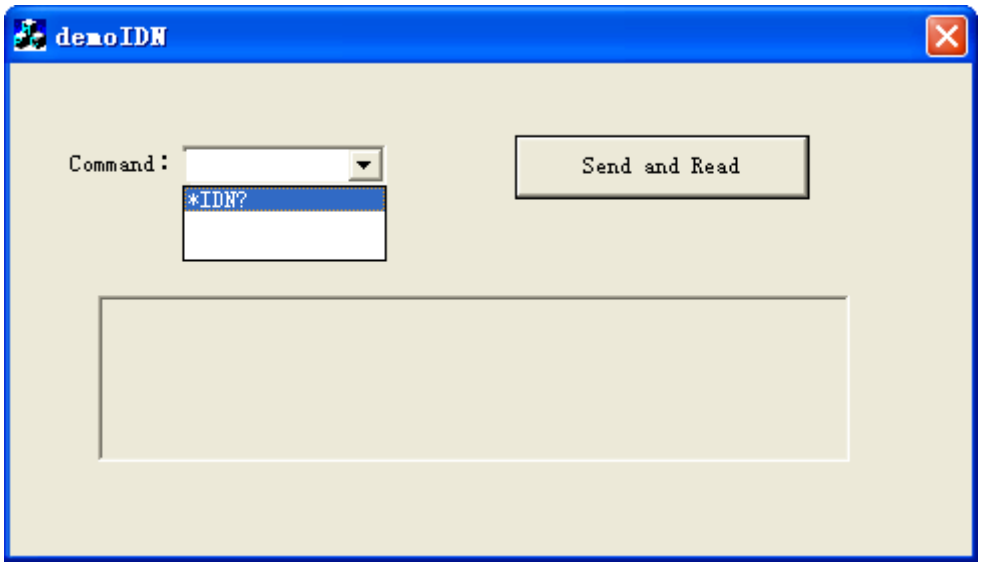
8. Add codes.

Double-click **Send and Read** to enter the programming environment. Declare the #include <visa.h> of the VISA library in the header file and then add the following codes:

```
ViSession defaultRM, vi;
char buf [256] = {0};
CString s,strTemp;
char* stringTemp;

ViChar buffer [VI_FIND_BUFLEN];
```

```
ViRsrc matches="buffer";
ViUInt32 nmatches;
ViFindList list;

viOpenDefaultRM (&defaultRM);
//Acquire the USB resource of VISA
viFindRsrc(defaultRM, "USB?*", &list,&nmatches, matches);
viOpen (defaultRM,matches,VI_NULL,VI_NULL,&vi);

//Send the command received
m_combox.GetLBText(m_combox.GetCurSel(),strTemp);
strTemp = strTemp + "\n";
stringTemp = (char *)(LPCTSTR)strTemp;
viPrintf (vi,stringTemp);

//Read the results
viScanf (vi, "%t\n", &buf);

//Display the results
UpdateData (TRUE);
m_receive = buf;
UpdateData (FALSE);
viClose (vi);
viClose (defaultRM);
```

9. Save, compile, and run the project to obtain a single exe file. When the instrument is correctly connected to the PC, enter a command (for example, *IDN?) and click **Send and Read** to execute the command. Then, the reading results will be returned.



HEADQUARTER

RIGOL TECHNOLOGIES CO., LTD.
No.8 Keling Road, New District, Suzhou,
JiangSu, P.R.China
Tel: +86-400620002
Email: info@rigol.com

EUROPE

RIGOL TECHNOLOGIES EU GmbH
Carl-Benz-Str.11
82205 Gilching
Germany
Tel: +49(0)8105-27292-0
Email: info-europe@rigol.com

NORTH AMERICA

RIGOL TECHNOLOGIES, USA INC.
10220 SW Nimbus Ave.
Suite K-7
Portland, OR 97223
Tel: +1-877-4-**RIGOL**-1
Fax: +1-877-4-**RIGOL**-1
Email: info@rigol.com

JAPAN

RIGOL JAPAN CO., LTD.
501, LATORRETTA, 2-37-1,
Numabukuro,
Nakano-Ku, Tokyo, Japan
Tel: +81-3-6262-8932
Fax: +81-3-6262-8933
Email: info-japan@rigol.com