RIGOL

DG70000 Series

Arbitrary Waveform Generator

Programming Guide

Aug.2022

Guaranty and Declaration

Copyright

Trademark Information

Notices

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 introduces how to program and control DG70000 series via the remote interface by using the SCPI (Standard Commands for Programmable Instruments) commands. DG70000 series can communicate with PC through USB, LAN, or GPIB interface (with the USB to GPIB interface module).

TIP

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

Publication Number

PGB13100-1110

Format Conventions in this Manual

1. Key

The front panel key is denoted by the menu key icon. For example, **Default** indicates the "Default" key.

2. Menu

The menu item is denoted by the format of "Menu Name (Bold) + Character Shading" in the manual, for example, **Setup**.

3. Operation Procedures

The next step of the operation is denoted by ">" in the manual. For example, ▦ > **Storage** indicates first clicking or tapping the icon ▦ and then clicking or tapping **Storage**.

Content Conventions in this Manual

DG70000 series Arbitrary Waveform Generator (AWG) includes the following models. Unless otherwise specified, this manual takes DG70004 as an example to illustrate the functions and operation methods of DG70000 series.

Model	Number of Channels	Maximum Output Frequency
DG70004	4	5 GHz

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 question 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,

:BWAVeform:AMP <*amp*>

:BWAVeform:AMP?

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

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

:SLISt:SEQuence:EVENt:JTIMing <*seq_name*>,<*type*>

Symbol Description

The following symbols are not sent with the commands.

1. **Braces { }**

   Braces are used to enclose one or more parameters, which 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,

`:CLOCk:OUTPut[:STATe]` <*bool*>

`:CLOCk:OUTPut[:STATe]?`

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

2. Discrete

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

`:BWAVeform:AUTO` <*param*>

`:BWAVeform:AUTO?`

Wherein,

- <*param*> can be set to LEN|CYCLe|DUR|FREQ|SRATe.
- The query returns LEN, CYCL, DUR, FREQ, or SRAT.

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,

`:BWAVeform:MTONe:COUNt` <*cnt*>

`:BWAVeform:MTONe:COUNt?`

Wherein, <*cnt*> can be set to an integer ranging from 2 to 16. The query returns an integer ranging from 2 to 16.

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,

`:TRIGger:INTerval` <*interval*>

:TRIGger:INTerval?

Wherein, <*interval*> can be set to any real number ranging from 1E-5 (10 μs) to 1E+1 (10s). The query returns a real number in scientific notation.

5. ASCII String

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

:WLISt:WAVeform:DELete <*wfm*>

Wherein, <*wfm*> can be set to Wave_1.

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,

:BWAVeform:OVERwrite?

can be abbreviated as

:BWAV:OVER?

3 AWG Commands

This chapter introduces the syntax, functions, parameters, and usage of each DG70000 (AWG) command.

CAUTION

1. **Unless otherwise specified, the descriptions in this manual take DG70004 as an example.**
2. **For the parameter setting command (time, frequency, amplitude, etc.), the instrument can only recognize the numbers, unable to recognize the unit sent together with them. For the default units of various parameters, refer to the descriptions for the specified command.**

3.1 :AWGControl Commands

:AWGControl commands are used to set or query the channel run state, set the active Strobe Edge (rising or falling edge) to use for Pattern Jump, and query the number of channels available on the instrument.

3.1.1 :AWGControl[:CHANnel[<n>]]:RSTate?

Syntax

`:AWGControl[:CHANnel[<n>]]:RSTate?`

Description

Queries the run state (Stopped or Playing) for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

If [:CHANnel[<n>]] or [<n>] is omitted, the command queries the run state of CH1 by default.

Return Format

The query returns STOP or RUN.

Examples

```
:AWGControl:CHANnel1:RSTate? /*Queries the run state of CH1. The
query returns RUN, indicating that CH1 is playing waveform.*/
```

3.1.2 :AWGControl[:CHANnel[<n>]]:RUN[:IMMediate]

Syntax

```
:AWGControl[:CHANnel[<n>]]:RUN[:IMMediate]
```

Description

Puts the specified channel in "Run" state. This is equivalent to pressing the RUN/STOP key on the front panel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

If [:CHANnel[<n>]] or [<n>] is omitted, the selected channel is CH1 by default.

Return Format

None.

Examples

```
:AWGControl:CHANnel1:RUN /*Puts the CH1 in run state.*/
```

3.1.3 :AWGControl[:CHANnel[<n>]]:STOP[:IMMediate]

Syntax

```
:AWGControl[:CHANnel[<n>]]:STOP[:IMMediate]
```

Description

Stops the playout of a waveform or sequence in the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

If [:CHANnel[<n>]] or [<n>] is omitted, the selected channel is CH1 by default.

Return Format

None.

Examples

```
:AWGControl:CHANnel1:STOP /*Stops the playout in CH1.*/
```

3.1.4 :AWGControl:CONFigure:CNUMber?

Syntax

:AWGControl:CONFigure:CNUMber?

Description

Queries the number of channels available on the instrument.

Parameter

None.

Remarks

None.

Return Format

The query returns 4.

Examples

```
:AWGControl:CONFigure:CNUMber? /*Queries the number of channels
available on the instrument. The query returns 4.*/
```

3.1.5 :AWGControl:PJUMp:SEDGe

Syntax

:AWGControl:PJUMp:SEDGe <*edge*>

:AWGControl:PJUMp:SEDGe?

Description

Sets or queries the active Strobe Edge (rising or falling edge) to use for Pattern Jump.

Parameter

Name	Type	Range	Default
<edge>	Discrete	{FALLing|RISing}	RISing

Remarks

- **FALLing** sets the pattern jump to occur on the falling edge of the strobe signal.
- **RISing** sets the pattern jump to occur on the rising edge of the strobe signal.

Return Format

The query returns FALL or RIS.

Examples

```
:AWGControl:PJUMp:SEDGe FALL /*Sets the pattern jump to occur on
the falling edge of the strobe signal.*/
:AWGControl:PJUMp:SEDGe? /*Queries the active Strobe Edge to use
for Pattern Jump. The query returns FALL.*/
```

3.2 :BWAVeform Commands

:BWAVeform commands are used to create and edit basic waveforms, multitone signals, and high-speed serial signals as well as convert the editing mode to Table Editor mode for the specified waveform.

3.2.1 :BWAVeform:AMP

Syntax

:BWAVeform:AMP <*amp*>

:BWAVeform:AMP?

Description

Sets or queries the Amplitude for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<amp>	Real	20 mVpp to 2 Vpp	500.0 mVpp

Remarks

Changing the Amplitude causes the High voltage value (*:BWAVeform:HIGH*), Low voltage value (*:BWAVeform:LOW*), and Offset (*:BWAVeform:OFFSet*) to adjust.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E+00, indicating that the amplitude is 1 Vpp.

Examples

```
:BWAVeform:AMP 1 /*Sets the Amplitude to 1 Vpp for the waveform
created by the Basic Waveform editor.*/
:BWAVeform:AMP? /*Queries the Amplitude for the waveform created by
the Basic Waveform editor. The query returns 1.0000000000E+00.*/
```

3.2.2 :BWAVeform:AUTO

Syntax

:BWAVeform:AUTO <*param*>

:BWAVeform:AUTO?

Description

Sets or queries the Calculate setting in Basic Waveform editor, determining which waveform property is automatically calculated.

Parameter

Name	Type	Range	Default
<param>	Discrete	{LEN|CYCLe|DUR|FREQ|SRATe}	LEN

Remarks

- The chosen property can only be calculated automatically based on other set properties.
- When the function is set to Sine, Square, Triangle, Sinc, or Sine (IQ), the options are LEN (Length), CYCLe (Cycle), FREQ (Frequency), and SRATe (Sample Rate). When the function is set to DC or Noise, the options are LEN (Length), DUR (Duration), and SRATe (Sample Rate).

Return Format

The query returns LEN, CYCL, DUR, FREQ, or SRAT.

Examples

```
:BWAVeform:AUTO LEN /*Sets the Length as the automatically
calculated value in Basic Waveform editor.*/
:BWAVeform:AUTO? /*Queries the Calculate setting in Basic Waveform
editor. The query returns LEN.*/
```

3.2.3 :BWAVeform:COMPile

Syntax

:BWAVeform:COMPile

Description

Initiates the compile operation in Basic Waveform editor. The created waveform is added in the waveform list.

Parameter

None.

Remarks

None.

Return Format

None.

Example

```
:BWAVeform:COMPile /*Compiles the basic waveform.*/
```

3.2.4 :BWAVeform:COMPile:CASSign

Syntax

`:BWAVeform:COMPile:CASSign` <*state*>

`:BWAVeform:COMPile:CASSign?`

Description

Sets or queries the state (enabled or disabled) of the Basic Waveform editor to either compile the waveform and immediately assign it to a specified channel (enabled) or just compile the waveform (disabled).

Parameter

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

Remarks

- 0 or OFF sets the Basic Waveform editor to compile only. 1 or ON sets the Basic Waveform editor to assign the waveform to a channel after compile.
- When using the assign function, you can send *:BWAVeform:COMPile:CHANnel* to set the playout channel intended for the compiled waveform.

Return Format

The query returns 0 or 1.

Examples

```
:BWAVeform:COMPile:CASSign 1 /*Enables the Basic Waveform editor to
assign the compiled waveform to a specified channel.*/
:BWAVeform:COMPile:CASSign? /*Queries whether to assign the
compiled waveform to a specified channel. The query returns 1.*/
```

3.2.5 :BWAVeform:COMPile:CHANnel

Syntax

:BWAVeform:COMPile:CHANnel <*chan*>

:BWAVeform:COMPile:CHANnel?

Description

Sets or queries the playout channel intended for the compiled waveform of the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<chan>	Discrete	{CH1|CH2|CH3|CH4|NONE}	-

Remarks

If the compile and assign function is enabled (*:BWAVeform:COMPile:CASSign*), the waveform will be assigned to the channel specified by this command when it is compiled (*:BWAVeform:COMPile*).

Return Format

The query returns CH1, CH2, CH3, CH4, or NONE.

Examples

```
:BWAVeform:COMPile:CHANnel CH1 /*Sets the playout channel intended
for the compiled waveform to CH1.*/
:BWAVeform:COMPile:CHANnel? /*Queries the playout channel intended
for the compiled waveform of the Basic Waveform editor.*/
```

3.2.6 :BWAVeform:COMPile:NAME

Syntax

:BWAVeform:COMPile:NAME <*name*>

:BWAVeform:COMPile:NAME?

Description

Sets or queries the name of the waveform to be compiled in Basic Waveform editor.

Parameter

Name	Type	Range	Default
<name>	ASCII string	Available waveform name	-

Remarks

- <name> specifies the name of the waveform which the *:BWAVeform:COMPile* command compiles and adds to the waveform list.
- You can use *:BWAVeform:OVERwrite* to set whether to overwrite the existing waveform of the same name.

Return Format

The query returns a string.

Examples

```
:BWAVeform:COMPile:NAME Wave /*Sets the name of the compiled
waveform to "Wave".*/
:BWAVeform:COMPile:NAME? /*Queries the name of the compiled
waveform. The query returns "Wave".*/
```

3.2.7 :BWAVeform:COMPile:PLAY

Syntax

:BWAVeform:COMPile:PLAY <*state*>

:BWAVeform:COMPile:PLAY?

Description

Sets or queries whether to immediately play the waveform after compile or just compile.

Parameter

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

Remarks

This command is available only when the compiled waveform is assigned to a channel (*:BWAVeform:COMPile:CHANnel*). 1 or ON enables the play after compile function.

Return Format

The query returns 0 or 1.

Examples

```
:BWAVeform:COMPile:PLAY 1 /*Enables the play after compile
function.*/
:BWAVeform:COMPile:PLAY? /*Queries whether the play after compile
function is enabled. The query returns 1.*/
```

3.2.8 :BWAVeform:CYCLe

Syntax

:BWAVeform:CYCLe <*cycle*>

:BWAVeform:CYCLe?

Description

Sets or queries the Cycles value (number of times the waveform repeats) for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<cycle>	Real	0.007813 to 644245094.4	2.4k

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.50000000000000E+03.

Examples

```
:BWAVeform:CYCLe 1500 /*Sets the Cycles value to 1500.*/
:BWAVeform:CYCLe? /*Queries the Cycles values. The query returns 
1.50000000000000E+03.*/
```

3.2.9 :BWAVeform:DURation

Syntax

:BWAVeform:DURation <*duration*>

:BWAVeform:DURation?

Description

Sets or queries the Duration for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<duration>	Real	480 ns to 16,106,127.36 s	2.4 μs

Remarks

The command is available only when the wave type (*:BWAVeform:FUNCtion*) is set to DC or Noise (NOISe).

Return Format

The query returns the duration in scientific notation. For example, the query may return 4.00000000000000E-06, indicating that the duration is 4 μs.

Examples

```
:BWAVeform:DURation 0.000004 /*Sets the Duration to 4 μs. The
default unit is s.*/
:BWAVeform:DURation? /*Queries the Duration. The query returns
4.00000000000000E-06.*/
```

3.2.10 :BWAVeform:DUTY

Syntax

:BWAVeform:DUTY <*duty*>

:BWAVeform:DUTY?

Description

Sets or queries the Duty Cycle for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<duty>	Real	0.001% to 99.999%	50%

Remarks

This command is available only when the wave type (*:BWAVeform:FUNCtion*) is set to square wave (SQUare).

Return Format

The query returns a real number. For example, the query might return 60.000000, indicating that the duty cycle is 60%.

Examples

```
:BWAVeform:DUTY 60 /*Sets the Duty Cycle to 60% for the waveform
(Square) created by the Basic Waveform editor.*/
:BWAVeform:DUTY? /*Queries the Duty Cycle for the waveform (Square)
created by the Basic Waveform editor. The query returns 60.000000.*/
```

3.2.11 :BWAVeform:FREQuency

Syntax

:BWAVeform:FREQuency <*frequency*>

:BWAVeform:FREQuency?

Description

Sets or queries the Frequency for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<frequency>	Real	1 μHz to 2 GHz	1 GHz

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.50000000000000E+09, indicating that the frequency is 1.5 GHz.

Examples

```
:BWAVeform:FREQuency 1500000000 /*Sets the Frequency to 1.5 GHz.*/
:BWAVeform:FREQuency? /*Queries the Frequency for the waveform
created by the Basic Waveform editor. The query returns
1.50000000000000E+09.*/
```

3.2.12 :BWAVeform:FUNCtion

Syntax

:BWAVeform:FUNCtion <*func*>

:BWAVeform:FUNCtion?

Description

Sets or queries the type of the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<func>	Discrete	{SINE|DC|SQUare|TRIangle|NOISe|SINC|SINIQ}	SINE

Remarks

- **SINE:** sinusoidal wave
- **DC:** direct current
- **SQUare:** square wave
- **TRIangle:** triangle wave
- **NOISe:** noise
- **SINC:** "Sinc" function wave
- **SINIQ:** sine wave (IQ)

Return Format

The query returns SINE, DC, SQU, TRI, NOIS, SINC, or SINIQ.

Examples

```
:BWAVeform:FUNCtion DC /*Sets the waveform type to DC.*/
:BWAVeform:FUNCtion? /*Queries the waveform type. The query returns
DC.*/
```

3.2.13 :BWAVeform:HIGH

Syntax

:BWAVeform:HIGH <*high*>

:BWAVeform:HIGH?

Description

Sets or queries the High voltage value for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<high>	Real	-1 V to 1 V	250 mV

Remarks

- The High voltage value is always greater than the Low voltage value (*:BWAVeform:LOW*): High value-Low value ≥ 20 mV.
- Changing High and Low voltage values (*:BWAVeform:LOW*) causes the Amplitude value (*:BWAVeform:AMP*) and Offset value (*:BWAVeform:OFFSet*) to adjust.

Return Format

The query returns the value in scientific notation. For example, the query might return 3.0000000000E-01, indicating that the high voltage value is 300 mV.

Examples

```
:BWAVeform:HIGH 0.3 /*Sets the High voltage value to 300 mV.*/
:BWAVeform:HIGH? /*Queries the High voltage value for the waveform
created by the Basic Waveform editor. The query returns
3.0000000000E-01.*/
```

3.2.14 :BWAVeform:HSSerial (Optional)

:BWAVeform:HSSerial commands are used to set or query the high speed serial signal parameters. The commands require that the High Speed Serial editor is installed.

3.2.14.1 :BWAVeform:HSSerial:BRATe

Syntax

:BWAVeform:HSSerial:BRATe <*brate*>

:BWAVeform:HSSerial:BRATe?

Description

Sets or queries the Bit Rate for the high speed serial signal.

Parameter

Name	Type	Range	Default
<brate>	Integer	100 Mbps to 2.5 Gbps	500 Mbps

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 2.0000000000E+08, indicating that the bit rate is 200 Mbps.

Examples

```
:BWAVeform:HSSerial:BRATe 200000000 /*Sets the Bit Rate to 200
Mbps. The default unit is bps.*/
:BWAVeform:HSSerial:BRATe?/*Queries the Bit Rate. The query
returns 2.0000000000E+08.*/
```

3.2.14.2 :BWAVeform:HSSerial:CODE

Syntax

:BWAVeform:HSSerial:CODE <*code*>

:BWAVeform:HSSerial:CODE?

Description

Sets or queries the user-defined bit sequence for the high speed serial signal.

Parameter

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

Remarks

- <code> is a binary string which cannot be longer than 128 bits.
- When the "Pattern" (*:BWAVeform:HSSerial:MODE*) is set to "User" (USER), you can use this command to define the bit sequence.

Return Format

The query returns a binary string, for example, 11001.

Examples

```
:BWAVeform:HSSerial:CODE 11001 /*Defines the bit sequence to
11001.*/
:BWAVeform:HSSerial:CODE? /*Queries the user-defined bit sequence.
The query returns 11001.*/
```

3.2.14.3 :BWAVeform:HSSerial:COMPile

Syntax

`:BWAVeform:HSSerial:COMPile`

Description

Compiles the high speed serial signal and adds it to the waveform list.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
:BWAVeform:HSSerial:COMPile /*Compiles the high speed serial
signal.*/
```

3.2.14.4 :BWAVeform:HSSerial:EDGE:ENABle

Syntax

:BWAVeform:HSSerial:EDGE:ENABle <*state*>

:BWAVeform:HSSerial:EDGE:ENABle?

Description

Sets or queries the state (enabled or not) of Step Response for high-speed serial signal.

Parameter

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

Remarks

After the Step Response is enabled, the system will process the signal based on the selected edge time type (*:BWAVeform:HSSerial:EDGE:TYPE*), rise time (*:BWAVeform:HSSerial:EDGE:RTIMe*) and fall time (*:BWAVeform:HSSerial:EDGE:FTIMe*).

Return Format

The query returns 1 or 0.

Examples

```
:BWAVeform:HSSerial:EDGE:ENABle 1 /*Enables the Step Response.*/
:BWAVeform:HSSerial:EDGE:ENABle? /*Queries whether the Step
Response is enabled. The query returns 1.*/
```

3.2.14.5 :BWAVeform:HSSerial:EDGE:FTIMe

Syntax

:BWAVeform:HSSerial:EDGE:FTIMe <*ftime*>

:BWAVeform:HSSerial:EDGE:FTIMe?

Description

Sets or queries the Fall Time for the high speed serial signal.

Parameter

Name	Type	Range	Default
<ftime>	Real	0.01 to 0.5	0.1

Remarks

- Based on the selected edge time type (*:BWAVeform:HSSerial:EDGE:TYPE*), the fall time can be defined as the time required for the amplitude of a symbol to drop from 90% to 10% or from 80% to 20%.
- It ranges from 0.01 UI to 0.5 UI. The unit interval (UI) is the time interval taken to transmit one bit, or the time taken in a data stream by each symbol. When <ftime> is set to 0.5, the fall time is 0.5 times (0.5 UI) the single symbol width.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000000E-01, indicating that the falling edge time is 0.5 UI.

Examples

```
:BWAVeform:HSSerial:EDGE:FTIMe 0.5 /*Sets the Fall Time to 0.5
UI.*/
:BWAVeform:HSSerial:EDGE:FTIMe? /*Queries the Fall Time. The query
returns 5.00000000000000E-01.*/
```

3.2.14.6 :BWAVeform:HSSerial:EDGE:RTIMe

Syntax

:BWAVeform:HSSerial:EDGE:RTIMe <*rtime*>

:BWAVeform:HSSerial:EDGE:RTIMe?

Description

Sets or queries the Rise Time for the high speed serial signal.

Parameter

Name	Type	Range	Default
<rtime>	Real	0.01 to 0.5	0.1

Remarks

- Based on the selected edge time type (*:BWAVeform:HSSerial:EDGE:TYPE*), the rise time can be defined as the time required for the amplitude of a symbol to rise from 10% to 90% or from 20% to 80%.
- It ranges from 0.01 UI to 0.5 UI. The unit interval (UI) is the time interval taken to transmit one bit, or the time taken in a data stream by each symbol. When <rtime> is set to 0.5, the rise time is 0.5 times (0.5 UI) the single symbol width.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000000E-01, indicating that the rising edge time is 0.5 UI.

Examples

```
:BWAVeform:HSSerial:EDGE:RTIMe 0.5 /*Sets the Rise Time to 0.5
UI.*/
:BWAVeform:HSSerial:EDGE:RTIMe? /*Queries the Rise Time. The query
returns 5.00000000000000E-01.*/
```

3.2.14.7 :BWAVeform:HSSerial:EDGE:TYPE

Syntax

:BWAVeform:HSSerial:EDGE:TYPE <*type*>

:BWAVeform:HSSerial:EDGE:TYPE?

Description

Sets or queries how the edge time (rise/fall time) is measured.

Parameter

Name	Type	Range	Default
<type>	Discrete	{10_90|20_80}	10_90

Remarks

- **20_80** sets the type to 20%~80%. You can set the rise time (*:BWAVeform:HSSerial:EDGE:RTIMe*) and fall time (*:BWAVeform:HSSerial:EDGE:FTIMe*) within this range.
- **10_90** sets the type to 10%~90%. You can set the rise time and fall time within this range.

Return Format

The query returns 10_90 or 20_80.

Examples

```
:BWAVeform:HSSerial:EDGE:TYPE 10_90 /*Sets the edge time to be
measured between 10% and 90%.*/
:BWAVeform:HSSerial:EDGE:TYPE? /*Queries how the edge time is
measured. The query returns 10_90.*/
```

3.2.14.8 :BWAVeform:HSSerial:FILE:PATH

Syntax

:BWAVeform:HSSerial:FILE:PATH <*path*>[,<*eol*>]

:BWAVeform:HSSerial:FILE:PATH?

Description

Sets or queries the path from which the pattern file is imported.

Parameter

Name	Type	Range	Default
<path>	ASCII string	Available path	-
<eol>	Discrete	{ENTer|COMMa|SEMicolon}	-

Remarks

When the "Pattern" (*:BWAVeform:HSSerial:MODE*) is set to "File" (FILE), you can use this command to set the path from which the pattern file is imported.

<eol> specifies the separator between each unit data.

- **ENTer** sets the separator to enter.
- **COMMa** sets the separator to comma ",".
- **SEMicolon** sets the separator to semicolon ";".

Return Format

The query returns an ASCII string, for example, D:/Wave.txt.

Examples

```
:BWAVeform:HSSerial:FILE:PATH D:/Wave.txt /*Sets the path to D:/
Wave.txt.*/
:BWAVeform:HSSerial:FILE:PATH? /*Queries the path from which the
pattern file is imported. The query returns D:/Wave.txt.*/
```

3.2.14.9 :BWAVeform:HSSerial:HIGH

Syntax

:BWAVeform:HSSerial:HIGH <*high*>

:BWAVeform:HSSerial:HIGH?

Description

Sets or queries the High Level for the high speed serial signal.

Parameter

Name	Type	Range	Default
<high>	Real	-0.975 V to 0.995 V	750 mV

Remarks

The High Level must be greater than the Low Level (*:BWAVeform:HSSerial:LOW*), and High Level-Low Level ≥ 20 mV.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E-01.

Examples

```
:BWAVeform:HSSerial:HIGH 0.5 /*Sets the High Level to 500 mV. The
default unit is V.*/
:BWAVeform:HSSerial:HIGH? /*Queries the High Level. The query
returns 5.0000000000E-01.*/
```

3.2.14.10 :BWAVeform:HSSerial:INVert

Syntax

:BWAVeform:HSSerial:INVert <*state*>

:BWAVeform:HSSerial:INVert?

Description

Sets or queries the state (enabled or not) of the Invert Bits function for high-speed serial signal.

Parameter

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

Remarks

None.

Return Format

The query returns 0 or 1.

Examples

```
:BWAVeform:HSSerial:INVert 1 /*Enables the Invert Bits function.*/
:BWAVeform:HSSerial:INVert? /*Queries whether the Invert Bits
function is enabled. The query returns 1.*/
```

3.2.14.11 :BWAVeform:HSSerial:KSRate

Syntax

:BWAVeform:HSSerial:KSRate <*state*>

:BWAVeform:HSSerial:KSRate?

Description

Sets or queries the state (enabled or not) of the Keep SR function for the high speed serial signal.

Parameter

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

Remarks

When the Keep SR function is enabled, the sample rate is fixed to 5 GHz.

Return Format

The query returns 1 or 0.

Examples

```
:BWAVeform:HSSerial:KSRate 1 /*Enables the Keep SR function.*/
:BWAVeform:HSSerial:KSRate? /*Queries whether the Keep SR function
is enabled. The query return 1.*/
```

3.2.14.12 :BWAVeform:HSSerial:LOW

Syntax

:BWAVeform:HSSerial:LOW <*low*>

:BWAVeform:HSSerial:LOW?

Description

Sets or queries the Low Level for the high speed serial signal.

Parameter

Name	Type	Range	Default
<low>	Real	-0.995 V to 0.975 V	-750 mV

Remarks

The Low Level must be less than the High Level (*:BWAVeform:HSSerial:HIGH*), and High Level-Low Level ≥ 20 mV.

Return Format

The query returns the value in scientific notation. For example, the query might return -5.0000000000E-01.

Examples

```
:BWAVeform:HSSerial:LOW -0.5 /*Sets the Low Level to -500 mV. The
default unit is V.*/
:BWAVeform:HSSerial:LOW? /*Queries the Low Level. The query
returns -5.0000000000E-01.*/
```

3.2.14.13 :BWAVeform:HSSerial:MODE

Syntax

:BWAVeform:HSSerial:MODE <*mode*>

:BWAVeform:HSSerial:MODE?

Description

Sets or queries the selected Pattern for the high speed serial signal.

Parameter

Name	Type	Range	Default
<mode>	Discrete	{CLK|ALL0|ALL1|PRBS|USER|FILE}	CLK

Remarks

- **CLK** sets the signal pattern to clock signal.
- **ALL0** sets the signal to a sequence in which all bits are 0.
- **ALL1** sets the signal to a sequence in which all bits are 1.
- **PRBS** sets the type to Pseudo Random Binary Sequence (PRBS). You can use *:BWAVeform:HSSerial:PRBS* to set the PRBS type.
- **USER** allows you to self-define the bit sequence. You can use *:BWAVeform:HSSerial:CODE* to set your desired sequence.
- **FILE** allows you to import user-defined codes via the external USB storage device.

Return Format

The query returns CLK, ALL0, ALL1, PRBS, USER, or FILE.

Examples

```
:BWAVeform:HSSerial:MODE USER /*Sets the Pattern to user-defined.*/
:BWAVeform:HSSerial:MODE? /*Queries the Pattern. The query returns
USER.*/
```

3.2.14.14 :BWAVeform:HSSerial:NOISe:ENABle

Syntax

:BWAVeform:HSSerial:NOISe:ENABle <*state*>

:BWAVeform:HSSerial:NOISe:ENABle?

Description

Sets or queries the state (enabled or not) of Noise for high-speed serial signal.

Parameter

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

Remarks

After the noise is enabled, the gaussian white noise is added to the signal based on the magnitude setting in *:BWAVeform:HSSerial:NOISe:MAGNitude* .

Return Format

The query returns 0 or 1.

Examples

```
:BWAVeform:HSSerial:NOISe:ENABle 1 /*Enables the Noise for high-
speed serial signal.*/
:BWAVeform:HSSerial:NOISe:ENABle? /*Queries whether the Noise is
enabled. The query returns 1.*/
```

3.2.14.15 :BWAVeform:HSSerial:NOISe:MAGNitude

Syntax

:BWAVeform:HSSerial:NOISe:MAGNitude <*magnitude*>

:BWAVeform:HSSerial:NOISe:MAGNitude?

Description

Sets or queries the noise Magnitude for the high speed serial signal.

Parameter

Name	Type	Range	Default
<magnitude>	Real	1.00 mV to 500.00 mV	10 mV

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.00000000000000E-02, indicating that the noise magnitude is 10 mV.

Examples

```
:BWAVeform:HSSerial:NOISe:MAGNitude 0.01 /*Sets the noise
Magnitude to 10 mV. The default unit is V.*/
:BWAVeform:HSSerial:NOISe:MAGNitude? /*Queries the noise
Magnitude. The query returns 1.00000000000000E-02.*/
```

3.2.14.16 :BWAVeform:HSSerial:PRBS

Syntax

:BWAVeform:HSSerial:PRBS <*prbs*>

:BWAVeform:HSSerial:PRBS?

Description

Sets or queries the PRBS type for the high speed serial signal.

Parameter

Name	Type	Range	Default
<prbs>	Discrete	{5|6|7|8|9|10|11|12|13|14|15|23}	5

Remarks

When the "Pattern" (*:BWAVeform:HSSerial:MODE*) is set to PRBS, you can use this command to set the PRBS pattern. The bit length for PRBSn is 2^n-1.

Return Format

The query returns 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, or 23.

Examples

```
:BWAVeform:HSSerial:PRBS 11 /*Sets the PRBS type to 11.*/
:BWAVeform:HSSerial:PRBS? /*Queries the PRBS type. The query
returns 11.*/
```

3.2.14.17 :BWAVeform:HSSerial:RJITter:ENABle

Syntax

:BWAVeform:HSSerial:RJITter:ENABle <*state*>

:BWAVeform:HSSerial:RJITter:ENABle?

Description

Sets or queries the state (enabled or not) of Random Jitter for high-speed serial signal.

Parameter

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

Remarks

"Random Jitter" is the deviation in the edge timing with respect to the ideal period. Turning on the random jitter adds jitter to signal based on the magnitude setting (*:BWAVeform:HSSerial:RJITter:MAGNitude*).

Return Format

The query returns 0 or 1.

Examples

```
:BWAVeform:HSSerial:RJITter:ENABle 1 /*Enables the Random Jitter.*/
:BWAVeform:HSSerial:RJITter:ENABle? /*Queries whether the Random
Jitter is enabled. The query returns 1.*/
```

3.2.14.18 :BWAVeform:HSSerial:RJITter:MAGNitude

Syntax

:BWAVeform:HSSerial:RJITter:MAGNitude <*magnitude*>

:BWAVeform:HSSerial:RJITter:MAGNitude?

Description

Sets or queries the jitter Magnitude for the high speed serial signal.

Parameter

Name	Type	Range	Default
<magnitude>	Real	0.01 to 0.5	0.01

Remarks

- "Random Jitter" is the deviation in the edge timing with respect to the ideal period.
- It ranges from 0.01 UI to 0.5 UI. The unit interval (UI) is the time interval taken to transmit one bit, or the time taken in a data stream by each symbol. When <magnitude> is set to 0.05, the random jitter magnitude is 0.05 times (0.05 UI) the single symbol width.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.00000000000000E-02, indicating that the magnitude is 0.05 UI.

Examples

```
:BWAVeform:HSSerial:RJITter:MAGNitude 0.05 /*Sets the Magnitude to
0.05 UI.*/
:BWAVeform:HSSerial:RJITter:MAGNitude? /*Queries the Magnitude.
The query returns 5.00000000000000E-02.*/
```

3.2.15 :BWAVeform:LENGth

Syntax

:BWAVeform:LENGth <*length*>

:BWAVeform:LENGth?

Description

Sets or queries the Length for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<length>	Integer	2400 to 1610612736	12000

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+04, indicating that the length is 15000.

Examples

```
:BWAVeform:LENGth 15000 /*Sets the Length to 15000.*/
:BWAVeform:LENGth? /*Queries the Length for the waveform created by
the Basic Waveform editor. The query returns 1.5000000000E+04.*/
```

3.2.16 :BWAVeform:LOOPwidth

Syntax

:BWAVeform:LOOPwidth <*loopwidth*>

:BWAVeform:LOOPwidth?

Description

Sets or queries the LoopWidth for the waveform (Sinc) created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<loopwidth>	Real	0.001% to 99.99%	50%

Remarks

<loopwidth> specifies the percentage of the loop in the entire length.

Return Format

The query returns a real number. For example, the query might return 60.000000, indicating that the loop width is 60%.

Examples

```
:BWAVeform:LOOPwidth 60 /*Sets the Loop Width 60.0%*/
:BWAVeform:LOOPwidth? /*Queries the Loop Width. The query returns
60.000000.*/
```

3.2.17 :BWAVeform:LOW

Syntax

:BWAVeform:LOW <*low*>

:BWAVeform:LOW?

Description

Sets or queries the Low voltage value for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<low>	Real	-1 V to 1 V	-250 mV

Remarks

- The High voltage value (*:BWAVeform:HIGH*) is always greater than the Low voltage value: High value-Low value ≥ 20 mV.
- Changing the High voltage value (*:BWAVeform:HIGH*) and Low voltage value causes the Amplitude value (*:BWAVeform:AMP*) and Offset value (*:BWAVeform:OFFSet*) to adjust.

Return Format

The query returns the value in scientific notation. For example, the query might return -3.0000000000E-01, indicating that the low voltage value is -300 mV.

Examples

```
:BWAVeform:LOW -0.3 /*Sets the Low voltage value to -300 mV.*/
:BWAVeform:LOW? /*Queries the Low voltage value for the waveform
created by the Basic Waveform editor. The query returns
-3.0000000000E-01.*/
```

3.2.18 :BWAVeform:MTONe (Optional)

:BWAVeform:MTONe commands are used to set or query multitone waveform parameters. The commands require that the Multitone & Chirp mode option is installed.

3.2.18.1 :BWAVeform:MTONe:COMPile

Syntax

`:BWAVeform:MTONe:COMPile`

Description

Compiles multitone and adds the waveform to the waveform list.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
:BWAVeform:MTONe:COMPile /*Compiles the multitone waveform.*/
```

3.2.18.2 :BWAVeform:MTONe:COUNt

Syntax

`:BWAVeform:MTONe:COUNt` <*cnt*>

`:BWAVeform:MTONe:COUNt?`

Description

Sets or queries the number of tones (Tone Count) in Tones mode.

Parameter

Name	Type	Range	Default
<cnt>	Integer	2 to 16	2

Remarks

The Tone Count and Spacing are mutually constrained. Tone Count=(End Frequency-Start Frequency)/Spacing. Both of them cannot be greater than the maximum value.

Return Format

The query returns an integer, for example, 3.

Examples

```
:BWAVeform:MTONe:COUNt 3 /*Sets the Tone Count to 3.*/
:BWAVeform:MTONe:COUNt? /*Queries the Tone Count. The query
returns 3.*/
```

3.2.18.3 :BWAVeform:MTONe:EFReq

Syntax

:BWAVeform:MTONe:EFReq <*freq*>

:BWAVeform:MTONe:EFReq?

Description

Sets or queries the End Frequency in Tones mode.

Parameter

Name	Type	Range	Default
<freq>	Real	2 kHz to 2 GHz	2 GHz

Remarks

- End Frequency ≥ Start Frequency (*:BWAVeform:MTONe:SFReq*)+1 kHz. If the input value is out of the range, it will be automatically modified by the system.
- The end frequency is limited by the sample rate (*:BWAVeform:MTONe:SRATe:RATE*). The end frequency is always less than or equal to sample rate/2.5.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+09, indicating that the end frequency is 1.5 GHz.

Examples

```
:BWAVeform:MTONe:EFReq 1500000000 /*Sets the End Frequency to 1.5
GHz.*/
:BWAVeform:MTONe:EFReq? /*Queries the End Frequency. The query
returns 1.5000000000E+09.*/
```

3.2.18.4 :BWAVeform:MTONe:FSTYpe

Syntax

:BWAVeform:MTONe:FSTYpe <*fstype*>

:BWAVeform:MTONe:FSTYpe?

Description

Sets or queries the frequency sweep in Chirp mode.

Parameter

Name	Type	Range	Default
<fstype>	Discrete	{LTHigh|HTLow}	LTHigh

Remarks

- **LTHigh:** sweeps from the low to high frequency within the sweep time.
- **HTLow:** sweeps from the high to low frequency within the sweep time.

Return Format

The query returns LTH or HTL.

Examples

```
:BWAVeform:MTONe:FSTYpe HTLow /*Sets the frequency sweep to High
to Low.*/
:BWAVeform:MTONe:FSTYpe? /*Queries the frequency sweep. The query
returns HTL.*/
```

3.2.18.5 :BWAVeform:MTONe:HFReq

Syntax

:BWAVeform:MTONe:HFReq <*freq*>

:BWAVeform:MTONe:HFReq?

Description

Sets or queries the High Frequency in Chirp mode.

Parameter

Name	Type	Range	Default
<freq>	Real	2 Hz to 2 GHz	1.1 GHz

Remarks

- High Frequency ≥ Low Frequency (*:BWAVeform:MTONe:LFReq*) +1 Hz. If the input value is out of the range, it will be automatically modified by the system.
- The high frequency is limited by the sample rate (*:BWAVeform:MTONe:SRATe:RATE*). The high frequency is always less than or equal to sample rate/2.5.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.8000000000E+09, indicating that the high frequency is 1.8 GHz.

Examples

```
:BWAVeform:MTONe:HFReq 1800000000 /*Sets the High Frequency in
Chirp mode to 1.8 GHz.*/
:BWAVeform:MTONe:HFReq? /*Queries the High Frequency in Chirp
mode. The query returns 1.8000000000E+09.*/
```

3.2.18.6 :BWAVeform:MTONe:LFReq

Syntax

:BWAVeform:MTONe:LFReq <*freq*>

:BWAVeform:MTONe:LFReq?

Description

Sets or queries the Low Frequency in Chirp mode.

Parameter

Name	Type	Range	Default
<freq>	Real	1 Hz to 1.999999999 GHz	1 GHz

Remarks

High Frequency (*:BWAVeform:MTONe:HFReq*) ≥ Low Frequency+1 Hz. If the input value is out of the range, it will be automatically modified by the system.

Return Format

The query returns the value in scientific notation. For example, the query might return 8.0000000000E+08, indicating that the low frequency is 800 MHz.

Examples

```
:BWAVeform:MTONe:LFReq 800000000/*Sets the Low Frequency in Chirp
mode to 800 MHz.*/
:BWAVeform:MTONe:LFReq? /*Queries the Low Frequency in Chirp mode.
The query returns 8.0000000000E+08.*/
```

3.2.18.7 :BWAVeform:MTONe:PHASe

Syntax

:BWAVeform:MTONe:PHASe <*phase*>

:BWAVeform:MTONe:PHASe?

Description

Sets or queries the Phase in Tones mode.

Parameter

Name	Type	Range	Default
<phase>	Real	0° to 360°	0°

Remarks

None.

Return Format

The query returns a real number, for example, 90.000000.

Examples

```
:BWAVeform:MTONe:PHASe 90 /*Sets the Phase to 90°.*/
:BWAVeform:MTONe:PHASe? /*Queries the Phase. The query returns
90.000000.*/
```

3.2.18.8 :BWAVeform:MTONe:SFReq

Syntax

:BWAVeform:MTONe:SFReq <*freq*>

:BWAVeform:MTONe:SFReq?

Description

Sets or queries the Start Frequency in Tones mode.

Parameter

Name	Type	Range	Default
<freq>	Real	1 kHz to 1.999999 GHz	1 GHz

Remarks

End Frequency (*:BWAVeform:MTONe:EFReq*) ≥ Start Frequency+1 kHz. If the input value is out of the range, it will be automatically modified by the system.

Return Format

The query returns the value in scientific notation. For example, the query might return 8.0000000000E+08, indicating that the start frequency is 800 MHz.

Examples

```
:BWAVeform:MTONe:SFReq 800000000 /*Sets the Start Frequency to 800
MHz.*/
:BWAVeform:MTONe:SFReq? /*Queries the Start Frequency. The query
returns 8.0000000000E+08.*/
```

3.2.18.9 :BWAVeform:MTONe:SPACing

Syntax

:BWAVeform:MTONe:SPACing <*spacing*>

:BWAVeform:MTONe:SPACing?

Description

Sets or queries the spacing between tones (Spacing) in Tones mode.

Parameter

Name	Type	Range	Default
<spacing>	Real	1 kHz to 1.999999 GHz	1 GHz

Remarks

The Spacing and Tone Count (*:BWAVeform:MTONe:COUNt*) are mutually constrained. Tone Count=(End Frequency-Start Frequency)/Spacing. Both of them cannot be greater than the maximum value.

Return Format

The query returns a real number. For example, the query might return 1500000000.000000, indicating that the spacing is 1.5 GHz.

Examples

```
:BWAVeform:MTONe:SPACing 1500000000 /*Sets the Spacing to 1.5
GHz.*/
:BWAVeform:MTONe:SPACing? /*Queries the Spacing. The query returns
1500000000.000000.*/
```

3.2.18.10 :BWAVeform:MTONe:SRATe:RATE

Syntax

:BWAVeform:MTONe:SRATe:RATE <*sample_rate*>

:BWAVeform:MTONe:SRATe:RATE?

Description

Sets or queries the Sample Rate of the Multitone signal.

Parameter

Name	Type	Range	Default
<sample_rate>	Real	5 kSa/s to 5 GSa/s	5 GSa/s

Remarks

- When setting the mode of sample rate (*:BWAVeform:MTONe:SRATe:TYPE*) to User Defined (USER), you can send this command to set the sample rate.
- In Chirp mode, the sample rate is always greater than or equal to 2.5 times the high frequency (*:BWAVeform:MTONe:HFReq*). In Tones mode, the sample rate is always greater than or equal to 2.5 times the end frequency (*:BWAVeform:MTONe:EFReq*). Changing the sample rate automatically modifies the high frequency and end frequency.

Return Format

The query returns the value in scientific notation. For example, the query might return 3.0000000000E+09, indicating that the sample rate is 3 GSa/s.

Examples

```
:BWAVeform:MTONe:SRATe:RATE 3000000000 /*Sets the Sample Rate to 3
GSa/s. The default unit is Sa/s.*/
:BWAVeform:MTONe:SRATe:RATE? /*Queries the Sample Rate. The query
returns 3.0000000000E+09.*/
```

3.2.18.11 :BWAVeform:MTONe:SRATe:TYPE

Syntax

:BWAVeform:MTONe:SRATe:TYPE <*type*>

:BWAVeform:MTONe:SRATe:TYPE?

Description

Sets or queries the sample rate mode of the Multitone signal.

Parameter

Name	Type	Range	Default
<type>	Discrete	{AUTO|USER}	AUTO

Remarks

- **AUTO** enables the Auto calculate mode. The sample rate is fixed to 5.00 GSa/s.
- **USER** allows you to self-define the sample rate which can range from 5 kSa/s to 5 GSa/s.

Return Format

The query returns AUTO or USER.

Examples

```
:BWAVeform:MTONe:SRATe:TYPE USER /*Sets the sample rate mode to
User Defined.*/
:BWAVeform:MTONe:SRATe:TYPE? /*Queries the sample rate mode. The
query returns USER.*/
```

3.2.18.12 :BWAVeform:MTONe:SWERate

Syntax

:BWAVeform:MTONe:SWERate <*sweep_rate*>

:BWAVeform:MTONe:SWERate?

Description

Sets or queries the Sweep Rate in Chirp mode.

Parameter

Name	Type	Range	Default
<sweep_rate>	Real	0 Hz/μs to 4 GHz/μs	10 MHz/μs

Remarks

The Sweep Time (*:BWAVeform:MTONe:SWETime*) and Sweep Rate are mutually constrained. Sweep Time=(High Frequency-Low Frequency)/Sweep Rate. Their values cannot exceed the range.

Return Format

The query returns a real number. For example, the query might return 20000000.000000, indicating that the sweep rate is 20 MHz/μs.

Examples

```
:BWAVeform:MTONe:SWERate 20000000 /*Sets the Sweep Rate to 20 MHz/
µs. The default unit is Hz/µs.*/
:BWAVeform:MTONe:SWERate? /*Queries the Sweep Rate. The query
returns 20000000.000000.*/
```

3.2.18.13 :BWAVeform:MTONe:SWETime

Syntax

:BWAVeform:MTONe:SWETime <*sweep_time*>

:BWAVeform:MTONe:SWETime?

Description

Sets or queries the Sweep Time in Chirp mode.

Parameter

Name	Type	Range	Default
<sweep_time>	Real	500 ns to 322122.5472 s	10 µs

Remarks

- You can send this command to set the Sweep Time. Based on that, the Sweep Rate is automatically calculated: Sweep Rate=(High Frequency-Low Frequency)/ Sweep Time.
- Sweep Time is affected by Sample Rate: 2500 ≤ Sweep Time*Sample Rate ≤ 1610612736.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E+00, indicating that the sweep time is 1 s.

Examples

```
:BWAVeform:MTONe:SWETime 1 / *Sets the Sweep Time to 1 s.*/
:BWAVeform:MTONe:SWETime? /*Queries the Sweep Time. The query
returns 1.0000000000E+00.*/
```

3.2.18.14 :BWAVeform:MTONe:TYPE

Syntax

:BWAVeform:MTONe:TYPE <*type*>

:BWAVeform:MTONe:TYPE?

Description

Sets or queries the type of the multitone signal (Tones or Chirp).

Parameter

Name	Type	Range	Default
<type>	Discrete	{TONes|CHIRp}	TONes

Remarks

- **TONes** selects the Tones mode to generate multitone waveforms composed of equally spaced tones at different frequencies.
- **CHIRp** selects the Chirp mode to generate linear chirp waveforms.

Return Format

The query returns TON or CHIR.

Examples

```
:BWAVeform:MTONe:TYPE CHIRp /*Sets the type to Chirp.*/
:BWAVeform:MTONe:TYPE? /*Queries the type. The query returns
CHIR.*/
```

3.2.19 :BWAVeform:OFFSet

Syntax

:BWAVeform:OFFSet <*offset*>

:BWAVeform:OFFSet?

Description

Sets or queries the Offset for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<offset>	Real	-1 V to 1 V	0 V

Remarks

Changing the Offset causes the instrument to recalculate the High voltage value (*:BWAVeform:HIGH*) and Low voltage value (*:BWAVeform:LOW*) without changing the Amplitude (*:BWAVeform:AMP*).

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-01, indicating that the offset is 100 mV.

Examples

```
:BWAVeform:OFFSet 0.1 /*Sets the Offset to 100 mV.*/
:BWAVeform:OFFSet? /*Queries the Offset. The query returns
1.0000000000E-01.*/
```

3.2.20 :BWAVeform:OVERwrite

Syntax

:BWAVeform:OVERwrite <*state*>

:BWAVeform:OVERwrite?

Description

Sets or queries whether the "Overwrite existing waveform" function is enabled.

Parameter

Name	Type	Range	Default
<state>	Discrete	{0|1|ON|OFF}	0

Remarks

1 or ON overwrites the existing waveform of the same name. 0 or OFF automatically adds the suffix "_n" (n is a positive integer) to the filename, for example, "Wave_1".

Return Format

The query returns 0 or 1.

Examples

```
:BWAVeform:OVERwrite 1 /*Enables the created waveform to overwrite
an existing waveform.*/
:BWAVeform:OVERwrite? /*Queries whether or not the created waveform
overwrites an existing waveform. The query returns 1.*/
```

3.2.21 :BWAVeform:PEAKpos

Syntax

:BWAVeform:PEAKpos <*peakpos*>

:BWAVeform:PEAKpos?

Description

Sets or queries the Peak Position for the waveform (Sinc) created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<peakpos>	Real	0.001% to 99.999%	50%

Remarks

<peakpos> specifies the position of the crest point on a wave.

Return Format

The query returns a real number between 0.001 and 99.999. For example, the query might return 30.000000.

Examples

```
:BWAVeform:PEAKpos 30 /*Sets the Peak Position for Sinc to 30%.*/
:BWAVeform:PEAKpos? /*Queries the Peak Position. The query returns
30.000000.*/
```

3.2.22 :BWAVeform:PHASe

Syntax

:BWAVeform:PHASe <*phase*>

:BWAVeform:PHASe?

Description

Sets or queries the Phase for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<phase>	Real	0º to 360º	0.000º

Remarks

None.

Return Format

The query returns a real number, for example, 94.200000.

Examples

```
:BWAVeform:PHASe 94.2 /*Sets the Phase to 94.2°.*/
:BWAVeform:PHASe? /*Queries the Phase. The query returns
94.200000.*/
```

3.2.23 :BWAVeform:RESet

Syntax

:BWAVeform:RESet

Description

Resets all parameters in Basic Waveform editor to their default values.

Parameter

None.

Remarks

None.

Return Format

None.

Example

```
:BWAVeform:RESet /*Sets all parameters in Basic Waveform editor to
default values.*/
```

3.2.24 :BWAVeform:SRATe

Syntax

:BWAVeform:SRATe <*sample_rate*>

:BWAVeform:SRATe?

Description

Sets or queries the Sample Rate for the waveform created by the Basic Waveform editor.

Parameter

Name	Type	Range	Default
<sample_rate>	Real	100 Sa/s to 5 GSa/s	5 GSa/s

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 3.2000000000E+09, indicating that the sample rate is 3.2 GSa/s.

Examples

```
:BWAVeform:SRATe 3200000000 /*Sets the Sample Rate to 3.2 GSa/s.*/
:BWAVeform:SRATe? /*Queries the Sample Rate. The query returns
3.2000000000E+09.*/
```

3.2.25 :BWAVeform:SYMM

Syntax

:BWAVeform:SYMM <*symm*>

:BWAVeform:SYMM?

Description

Sets or queries the Symmetry for the waveform (Triangle) created by the Basic Waveform editor.

Parameter

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

Remarks

None.

Return Format

The query returns a real number, for example, 50.000000.

Examples

```
:BWAVeform:SYMM 42.3 /*Sets the Symmetry to 42.3%.*/
:BWAVeform:SYMM? /*Queries the Symmetry. The query returns
42.300000.*/
```

3.2.26 :BWAVeform:TABLe:EDIT

Syntax

:BWAVeform:TABLe:EDIT <*edit*>

Description

Converts the editing mode to Table Editor mode for the specified waveform.

Parameter

Name	Type	Range	Default
<edit>	ASCII String	Existing waveform name	-

Remarks

Switching to the Table Editor mode allows you to modify any sample data in the waveform.

Return Format

None.

Examples

```
:BWAVeform:TABLe:EDIT Wave /*Converts the editing mode to Table
Editor mode for the waveform named "wave".*/
```

3.3 :CLOCk Commands

:CLOCk commands are used to set and query information related to the clock.

3.3.1 :CLOCk:ECLock:DIVider

Syntax

:CLOCk:ECLock:DIVider <*div*>

:CLOCk:ECLock:DIVider?

Description

Sets or queries the divider rate for the external clock.

Parameter

Name	Type	Range	Default
<div>	Integer	Refer to *Remarks*	1

Remarks

- <div> is 2^n (n=0,1,2,3,...), in which "n" depends on the external sample clock frequency.
- This command is valid only when the clock source (*:CLOCk:SOURce*) is set to external sample clock (EXTernal).

Return Format

The query returns an integer, for example, 4.

Examples

```
:CLOCk:ECLock:DIVider 4 /*Sets the external clock divider rate to
4.*/
:CLOCk:ECLock:DIVider? /*Queries the external clock divider rate.
The query returns 4.*/
```

3.3.2 :CLOCk:ECLock:FREQuency

Syntax

:CLOCk:ECLock:FREQuency <*freq*>

:CLOCk:ECLock:FREQuency?

Description

Sets or queries the expected frequency of the external clock.

Parameter

Name	Type	Range	Default
<freq>	Real	2.5 GHz to 6 GHz	5 GHz

Remarks

The command is valid only when the clock source (*:CLOCk:SOURce*) is set to external sample clock (EXTernal).

Return Format

The query returns the frequency in scientific notation. For example, the query might return 3.5000000000E+09, indicating that the expected frequency of the signal is 3.5 GHz.

Examples

```
:CLOCk:ECLock:FREQuency 3500000000 /*Sets the expected frequency of
the external clock to 3.5 GHz. The default unit is Hz.*/
:CLOCk:ECLock:FREQuency? /*Queries the expected frequency of the
external clock. The query returns 3.5000000000E+09.*/
```

3.3.3 :CLOCk:ECLock:FREQuency:DETect?

Syntax

:CLOCk:ECLock:FREQuency:DETect?

Description

Detects the frequency of the external sample clock.

Parameter

None.

Remarks

- This command detects the frequency of the signal applied to the **[SCLK IN]** connector. The frequency is detected once each time the command executes. An error message is generated if no frequency is detected or is out of range.

- This command is valid only when the clock source (*:CLOCk:SOURce*) is set to external sample clock (EXTernal).

Return Format

The query returns the frequency of the signal applied to the **[SCLK IN]** connector in scientific notation.

Examples

```
:CLOCk:ECLock:FREQuency:DETect? /*The query returns the frequency
of the external sample clock. The query returns 3.5000000000E+09.*/
```

3.3.4 :CLOCk:EREFerence:FREQuency

Syntax

:CLOCk:EREFerence:FREQuency <*freq*>

:CLOCk:EREFerence:FREQuency?

Description

Sets or queries the expected frequency of the external reference clock.

Parameter

Name	Type	Range	Default
<freq>	Real	35 MHz to 150 MHz	35 MHz

Remarks

This command is valid only when the clock source (*:CLOCk:SOURce*) is set to external variable reference clock (EVARiable).

Return Format

The query returns the frequency in scientific notation. For example, the query might return 1.0000000000E+08, indicating that the expected frequency of the signal is 100 MHz.

Examples

```
:CLOCk:EREFerence:FREQuency 100000000 /*Sets the expected frequency
of the external reference clock to 100 MHz.*/
:CLOCk:EREFerence:FREQuency? /*Queries the expected frequency of
the external reference clock. The query returns 1.0000000000E+08.*/
```

3.3.5 :CLOCk:EREFerence:FREQuency:DETect?

Syntax

:CLOCk:EREFerence:FREQuency:DETect?

Description

Detects the frequency of the external reference clock.

Parameter

None.

Remarks

- This command detects the frequency of the signal applied to the **[EXT REF IN]** connector. The frequency is detected once each time the command executes. An error message is generated if no frequency is detected or is out of range.
- This command is valid only when the clock source (*:CLOCk:SOURce*) is set to external variable reference clock (EVARiable).

Return Format

The query returns the frequency of the external reference clock in scientific notation.

Examples

```
:CLOCk:EREFerence:FREQuency:DETect? /*Queries the frequency of the
external reference clock. The query might return 1.0000000000E+08.*/
```

3.3.6 :CLOCk:INCLock:COMPlex

Syntax

:CLOCk:INCLock:COMPlex <*state*>

:CLOCk:INCLock:COMPlex?

Description

Sets or queries whether the internal clock frequency is within the frequency band in IQ mode.

Parameter

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

Remarks

- This command is valid only when you divide 5 GHz-6 GHz by 2^n to get the sample rate (*:CLOCk:SRATe*).
- When setting <state> to 1 or ON, you can send this command to set the internal sample clock frequency in the range of 5 GHz to 6 GHz (a frequency band available only when in IQ mode). For example, when the sample rate is set to 2.8

GSa/s, you can send :CLOCk:INCLock:COMPlex 1 to set the sample clock frequency to 5.6 GHz.

Return Format

The query returns 0 or 1.

Examples

```
:CLOCk:INCLock:COMPlex 1 /*Sets the internal clock frequency to
range within the frequency band in IQ mode.*/
:CLOCk:INCLock:COMPlex? /*Queries whether the internal clock
frequency is within the frequency band in IQ mode. The query
returns 1.*/
```

3.3.7 :CLOCk:OUTPut:FREQuency?

Syntax

:CLOCk:OUTPut:FREQuency?

Description

Queries the frequency of the Sample Clock Output.

Parameter

None.

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 3.5000000000E+09, indicating that the sample clock output frequency is 3.5 GHz.

Examples

```
:CLOCk:OUTPut:FREQuency? /*Queries the frequency of the Sample
Clock Output. The query returns 3.5000000000E+09.*/
```

3.3.8 :CLOCk:OUTPut[:STATe]

Syntax

:CLOCk:OUTPut[:STATe] <*state*>

:CLOCk:OUTPut[:STATe]?

Description

Sets or queries the on/off state of the clock output.

Parameter

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

Remarks

When the clock output is enabled, the instrument will directly output the system sample clock signal via the **[SCLK OUT]** connector on the rear panel.

Return Format

The query returns 0 or 1.

Examples

```
:CLOCk:OUTPut:STATe 1 /*Enables the clock output.*/
:CLOCk:OUTPut:STATe? /*Queries the on/off state of the clock
output. The query returns 1.*/
```

3.3.9 :CLOCk:SOURce

Syntax

:CLOCk:SOURce <*source*>

:CLOCk:SOURce?

Description

Sets or queries the current source of the clock.

Parameter

Name	Type	Range	Default
<source>	Discrete	{INTernal|EXTernal|EFIXed|EVARiable}	INTernal

Remarks

- **INTernal:** The clock is derived from the instrument's internal oscillator as the reference signal.
- **EXTernal:** The instrument directly accepts the signal applied to the rear-panel **[SCLK IN]** connector as the sample clock.
- **EFIXed:** The clock is derived from a fixed 10 MHz reference supplied at the **[EXT REF IN]** connector.
- **EVARiable:** The clock is derived from a variable reference supplied at the **[EXT REF IN]** connector. This reference frequency must be between 35 MHz and 150 MHz.

Return Format

The query returns INT, EXT, EFIX, or EVAR.

Examples

```
:CLOCk:SOURce INTernal /*Sets the clock source to internal
reference clock.*/
:CLOCk:SOURce? /*Queries the clock source. The query returns INT.*/
```

3.3.10 :CLOCk:SOUT[:STATe]

Syntax

:CLOCk:SOUT[:STATe] <*state*>

:CLOCk:SOUT[:STATe]?

Description

Sets or queries the on/off state of the Sync Clock Out output.

Parameter

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

Remarks

When the Sync Clock Out is enabled, the instrument will output sync clock signal via the **[SYNC OUT]** connector on the rear panel. The Sync Clock Out frequency is 1/32th of the Sample Clock frequency.

Return Format

The query returns 0 or 1.

Examples

```
:CLOCk:SOUT:STATe 1 /*Enables the Sync Clock Out output.*/
:CLOCk:SOUT:STATe? /*Queries the on/off state of the Sync Clock Out
output. The query returns 1.*/
```

3.3.11 :CLOCk:SRATe

Syntax

:CLOCk:SRATe <*rate*>

:CLOCk:SRATe?

Description

Sets or queries the sample rate for the clock.

Parameter

Name	Type	Range	Default
<rate>	Integer	100 Sa/s to 5 GSa/s	5 GSa/s

Remarks

- The command is not available when the clock source (*:CLOCk:SOURce*) is set to external sample clock (EXTernal).
- When using this command to set the sample rate by dividing 5 GHz-6 GHz by 2^n, you can use *:CLOCk:INCLock:COMPlex* to set the sample clock frequency in the range of 5 GHz to 6 GHz (a frequency band only available when in IQ mode).

Return Format

The query returns the sample rate in scientific notation. For example, the query might return 3.0000000000E+09, indicating that the clock sample rate is set to 3 GSa/s.

Examples

```
:CLOCk:SRATe 3000000000 /*Sets the clock sample rate to 3 GSa/s.*/
:CLOCk:SRATe? /*Queries the clock sample rate. The query returns
3.0000000000E+09.*/
```

3.4 :DISPlay Commands

:DISPlay commands are used to query the data stream of the current screen image.

3.4.1 :DISPlay:DATA?

Syntax

:DISPlay:DATA? [<*type*>]

Description

Queries the data stream of the current screen image.

Parameter

Name	Type	Range	Default
<type>	Discrete	{BMP|PNG|JPG}	PNG

Remarks

<type> specifies the format of the screen image whose binary data stream will be returned.

Return Format

The query returns the binary data stream of the screen image in the specified format.

Examples

```
None.
```

3.5 IEEE488.2 Common Commands

3.5.1 *CLS

Syntax

`*CLS`

Description

Clears all event registers and error queues.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
None.
```

3.5.2 *ESE

Syntax

`*ESE` <*maskargument*>

`*ESE?`

Description

Sets or queries the enabled bit in the enable register of the Standard Event register.

Parameter

Name	Type	Range	Default
<maskargument>	Integer	0 to 255	0

Remarks

- The <maskargument> 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.
- When it is set to 0, executing this command will clear the enable register of the Standard Event register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of all bits in the register. For example, the query might return 16.

Examples

```
*ESE 16 /*Enables bit4 (decimal 16) in the enable register of the
Standard Event register.*/
*ESE? /*Queries the enabled bit in the enable register of the
Standard Event register. The query returns 16.*/
```

3.5.3 *ESR?

Syntax

`*ESR?`

Description

Queries the decimal value of the binary-weighted sum of all bits in the event register of the Standard Event register and clears the value.

Parameter

None.

Remarks

The event register of the Standard Event register is a read-only register. Bits in it are latched. Querying the register will clear it. Once an event bit is set, subsequent state changes are ignored. It is automatically cleared by a query of that register or by sending the clear status command (**CLS*).

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of all bits in the register. For example, the query might return 16.

Examples

```
*ESR? /*Queries the decimal value of the binary-weighted sum of all
bits in the event register of the Standard Event register and
clears the value. The query returns 16.*/
```

3.5.4 *IDN?

Syntax

```
*IDN?
```

Description

Queries the instrument's identification string.

Parameter

None.

Remarks

None.

Return Format

The query returns the string in the format of RIGOL TECHNOLOGIES,<model>,<serial number>,<software version>.

- **<model>** specifies the model number.
- **<serial number>** specifies the serial number.
- **<software version>** specifies the software version.

Examples

```
None.
```

3.5.5 *OPC

Syntax

```
*OPC
*OPC?
```

Description

Sets the OPC (bit 0, "Operation Complete") of the Standard Event register to 1 after all commands are executed.

Queries whether all the previous commands are executed. The query returns 1 to the output buffer after the command is executed.

Parameter

None.

Remarks

- Operation complete means that all the previous commands including the *OPC command have been executed.
- When setting the instrument configuration through programming (by executing the command string), using this command as the last command can determine when the command queue is executed (when the command queue is executed, the bit0 (OPC, "operation complete" bit) in the event register of the Standard Event register will be set).
- Sending the *OPC? command and reading the result can ensure synchronization.

Return Format

Queries whether all the previous commands are executed. The query returns 1 when all commands are executed.

Examples

```
None.
```

3.5.6 *RCL

Syntax

`*RCL`

Description

Recalls a previously stored value in *SAV command from the specified location.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
None.
```

3.5.7 *RST

Syntax

`*RST`

Description

Resets the instrument to its power-on default state.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
None.
```

3.5.8 *SRE

Syntax

`*SRE` <*maskargument*>

`*SRE?`

Description

Sets or queries the bits in the enable register of the Status Byte register.

Parameter

Name	Type	Range	Default
<maskargument>	Integer	0 to 255	0

Remarks

- The <maskargument> 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. Accumulate the specified bits in bit6 of the Status Byte register. If one of the bit is set to 1 from 0, a service request occurs.
- When <maskargument> is set to 0, executing this command will clear the enable register of the Status Byte register.

Return Format

The query returns a decimal integer, which corresponds to the binary-weighted sum of all bits set in the register.

Examples

```
*SRE 16 /*Enables bit4 (decimal 16) in the enable register of the
Status Byte register.*/
*SRE? /*Queries the enabled bit in the register. The query returns
16.*/
```

3.5.9 *STB?

Syntax

***STB?**

Description

Queries the bits in the event 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 in the register). This command cannot clear service request. As long as the condition that generates the service request remains, it does not clear bit6 (Master Status Summary bit) in the Status Byte register.

Return Format

The query returns a decimal value, which corresponds to the binary-weighted sum of all bits in the register.

Examples

```
None.
```

3.6 :INSTrument Commands

:INSTrument commands are used to set and query the coupled state of analog outputs and marker outputs, or the opeartion mode of the generator.

3.6.1 :INSTrument:COUPle:SOURce

Syntax

:INSTrument:COUPle:SOURce <*value1*>[,<*value2*>[,<*value3*>[,<*value4*>]]]

:INSTrument:COUPle:SOURce?

Description

Sets or queries the coupled state of the channel's analog and marker outputs.

Parameter

Name	Type	Range	Default
<value1>	Discrete	{CH1|CH2|CH3|CH4|NONE|ALL}	-
<value2>	Discrete	{CH1|CH2|CH3|CH4}	-
<value3>	Discrete	{CH1|CH2|CH3|CH4}	-
<value4>	Discrete	{CH1|CH2|CH3|CH4}	-

Remarks

- You can select 2 to 4 channels from CH1 to CH4 to couple their settings. For example, you can set CH1,CH2.
- NONE sets no coupling.
- ALL couples all the four channels.

Return Format

The query returns a string. For example, the query might return CH1,CH2, indicating that CH1 and CH2 are coupled.

Examples

```
:INSTrument:COUPle:SOURce CH1,CH2 /*Couples CH1 and CH2.*/
:INSTrument:COUPle:SOURce? /*Queries the coupled channels. The
query returns CH1,CH2.*/
```

3.6.2 :INSTrument:MODE

Syntax

:INSTrument:MODE <*mode*>

:INSTrument:MODE?

Description

Sets or queries the operation mode of the generator, either AWG or AFG mode.

Parameter

Name	Type	Range	Default
<mode>	Discrete	{AWG|AFG}	AWG

Remarks

None.

Return Format

The query returns AWG or AFG.

Examples

```
:INSTrument:MODE AWG /*Sets the operation mode to AWG.*/
:INSTrument:MODE? /*Queries the operation mode of the generator.
The query returns AWG.*/
```

3.7 :LAN Commands

:LAN commands are used to set or query network and interface configuration.

3.7.1 :LAN:APPLy

Syntax

:LAN:APPLy

Description

Applies the network settings.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
:LAN:APPLy /*Applies the current network configuration.*/
```

3.7.2 :LAN:AUToip

Syntax

:LAN:AUToip <*bool*>

:LAN:AUToip?

Description

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

Parameter

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

Remarks

- The DHCP mode takes precedence over the Auto IP mode. Therefore, to make the Auto IP mode valid, DHCP should be disabled.
- In Auto IP mode, it obtains the IP address (169.254.0.1 to 169.254.255.254) and subnet mask (255.255.0.0) depending on the current network configuration.

Return Format

The query returns 0 or 1.

Examples

```
:LAN:AUToip OFF /*Disables the Auto IP mode.*/
:LAN:AUToip? /*Queries the on/off state of Auto IP. The query
returns 0.*/
```

3.7.3 :LAN:DESCription

Syntax

:LAN:DESCription <*name*>

:LAN:DESCription?

Description

Sets or queries the instrument description.

Parameter

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

Remarks

None.

Return Format

The query returns a string.

Examples

```
:LAN:DESCription RIGOL Arbitrary Waveform Generator /*Sets the
instrument description to RIGOL Arbitrary Waveform Generator.*/
:LAN:DESCription? /*Queries the instrument description. The query
returns RIGOL Arbitrary Waveform Generator.*/
```

3.7.4 :LAN:DHCP

Syntax

:LAN:DHCP <*bool*>

:LAN:DHCP?

Description

Sets or queries the on/off state of DHCP.

Parameter

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

Remarks

- When all the three configuration modes (DHCP, Auto IP, and Static IP) are enabled, the priority is "DHCP", "Auto IP", and "Static IP". You cannot turn off all of them at the same time.
- When DHCP is turned on, it automatically assigns an IP address to the instrument from a DHCP server.
- You need to send *:LAN:APPLy* to apply the settings.

Return Format

The query returns 0 or 1.

Examples

```
:LAN:DHCP OFF /*Disables the DHCP mode.*/
:LAN:DHCP? /*Queries the on/off state of DHCP. The query returns
0.*/
```

3.7.5 :LAN:DNS

Syntax

:LAN:DNS <*string*>

:LAN:DNS?

Description

Sets or queries the domain name server (DNS) address.

Parameter

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

Remarks

- The format of <string> is nnn.nnn.nnn.nnn; wherein, the first nnn ranges from 0 to 223 (excluding 127), and the other three range from 0 to 255.
- When using this command to set the DNS address, you should enable the Static IP mode and disable the DHCP and Auto IP mode.

Return Format

The query returns the DNS address in string, for example, 192.168.1.1.

Examples

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

3.7.6 :LAN:DSERver?

Syntax

`:LAN:DSERver?`

Description

Queries the DHCP server address.

Parameter

None.

Remarks

None.

Return Format

The query returns the DHCP server address in string.

Examples

```
:LAN:DSERver? /*Queries the DHCP server address.*/
```

3.7.7 :LAN:GATeway

Syntax

:LAN:GATeway <*string*>

:LAN:GATeway?

Description

Sets or queries the gateway.

Parameter

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

Remarks

- The format of <string> is nnn.nnn.nnn.nnn; wherein, the first nnn ranges from 0 to 223 (excluding 127), and the other three range from 0 to 255.
- When using this command to set the gateway, you should enable the Static IP mode and disable the DHCP and Auto IP mode.

Return Format

The query returns the gateway in string, for example, 192.168.1.1.

Examples

```
:LAN:GATeway 192.168.1.1 /*Sets the gateway to 192.168.1.1.*/
:LAN:GATeway? /*Queries the gateway. The query returns
192.168.1.1.*/
```

3.7.8 :LAN:GPIB

Syntax

:LAN:GPIB <*adr*>

:LAN:GPIB?

Description

Sets or queries the instrument's GPIB address.

Parameter

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

Remarks

None.

Return Format

The query returns an integer, for example, 12.

Examples

```
:LAN:GPIB 12 /*Sets the GPIB address to 12.*/
:LAN:GPIB? /*Queries the GPIB address. The query returns 12.*/
```

3.7.9 :LAN:HOST:NAME

Syntax

:LAN:HOST:NAME <*name*>

:LAN:HOST:NAME?

Description

Sets or queries the host name.

Parameter

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

Remarks

None.

Return Format

The query returns the host name in ASCII string.

Examples

```
:LAN:HOST:NAME Rigol StationMax /*Sets the host name to Rigol
StationMax.*/
:LAN:HOST:NAME? /*Queries the host name. The query returns Rigol
StationMax.*/
```

3.7.10 :LAN:IPADdress

Syntax

:LAN:IPADdress <*string*>

:LAN:IPADdress?

Description

Sets or queries the instrument's IP address.

Parameter

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

Remarks

- The format of <string> is nnn.nnn.nnn.nnn; wherein, the first nnn ranges from 0 to 223 (excluding 127), and the other three range from 0 to 255.
- When using this command to set the IP address, you should enable the Static IP mode and disable the DHCP and Auto IP mode.

Return Format

The query returns the IP address in string, for example, 192.168.1.10.

Examples

```
:LAN:IPADdress 192.168.1.10 /*Sets the IP address to 192.168.1.10.*/
:LAN:IPADdress? /*Queries the IP address. The query returns
192.168.1.10.*/
```

3.7.11 :LAN:MAC?

Syntax

:LAN:MAC?

Description

Queries the instrument's MAC address.

Parameter

None.

Remarks

None.

Return Format

The query returns the MAC address in string, for example, f6:e6:b4:c0:7f:62. For an instrument, the MAC address is always unique.

Examples

```
:LAN:MAC? /*Queries the instrument's MAC address.*/
```

3.7.12 :LAN:MANual

Syntax

:LAN:MANual <*bool*>

:LAN:MANual?

Description

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

Parameter

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

Remarks

- The DHCP and Auto IP modes take precedence over the Static IP mode. Therefore, to make the Static IP mode valid, DHCP and Auto IP should be disabled.
- When Static IP is valid, you can self-define the instrument's network parameters including IP address (*:LAN:IPADdress*), subnet mask (*:LAN:SMASk*), gateway (*:LAN:GATeway*), and DNS (*:LAN:DNS*).

Return Format

The query returns 0 or 1.

Examples

```
:LAN:MANual ON /*Turns on Static IP.*/
:LAN:MANual? /*Queries the on/off state of Static IP. The query
returns 1.*/
```

3.7.13 :LAN:MDNS

Syntax

:LAN:MDNS <*bool*>

:LAN:MDNS?

Description

Sets or queries the on/off state of the multicast Domain Name System (mDNS).

Parameter

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

Remarks

The multicast DNS protocol resolves hostnames to IP addresses within small networks that do not include a local name server.

Return Format

The query returns 1 or 0.

Examples

```
:LAN:MDNS ON /*Turns on the mDNS.*/
:LAN:MDNS? /*Queries the on/off state of the mDNS. The query
returns 1.*/
```

3.7.14 :LAN:SMASk

Syntax

:LAN:SMASk <*string*>

:LAN:SMASk?

Description

Sets or queries the subnet mask.

Parameter

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

Remarks

- The format of <string> is nnn.nnn.nnn.nnn; wherein, the range of the nnn is from 0 to 255.
- When using this command to set the subnet mask, you should enable the Static IP mode and disable the DHCP and Auto IP mode.

Return Format

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

Examples

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

3.7.15 :LAN:STATus?

Syntax

:LAN:STATus?

Description

Queries the current network configuration status.

Parameter

None.

Remarks

- **UNLINK:** no connection
- **CONNECTED:** connected
- **INIT:** obtaining IP
- **IPCONFLICT:** IP conflict
- **BUSY:** network busy
- **CONFIGURED:** successful network configuration
- **DHCPFAILED:** DHCP configuration failure
- **INVALIDIP:** invalid IP address
- **IPLOSE:** lose IP address

Return Format

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

Examples

```
:LAN:STATus? /*Queries the current network configuration status.*/
```

3.7.16 :LAN:VISA?

Syntax

:LAN:VISA? [<*type*>]

Description

Queries the instrument's VISA address.

Parameter

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

Remarks

VISA address describes the accurate name and location of the VISA resource. <type> specifies the address type to be queried.

- **USB** queries the USB address.
- **LXI** queries the Ethernet address.
- **SOCKET** queries the GPIB address.

Return Format

The query returns the VISA address in string.

Examples

```
:LAN:VISA? LXI /*Queries the Ethernet address. The query returns
TCPIP::172.18.10.6::INSTR.*/
```

3.8 :MODulation Commands

:MODulation commands are used to set or query information related to modulation.

3.8.1 :MODulation:COMPile

Syntax

```
:MODulation:COMPile
```

Description

Compiles the modulated waveforms.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
:MODulation:COMPile /*Compiles the modulated waveforms.*/
```

3.8.2 :MODulation:DATA:I?

Syntax

:MODulation:DATA:I? <*wfm_name*>[,<*startIndex*>[,<*size*>]]

Description

Queries the I data of the modulated waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-
<startIndex>	Integer	Refer to *Remarks*	-
<size>	Integer	Refer to *Remarks*	-

Remarks

- You can use this command to query a total number of <size> points starting from <startIndex>. startIndex+size-1 ≤ wave length. The minimum <size> is 1. If [<size>] is omitted, the length of waveform is assumed to be the value of the <size> parameter.
- You can use *:MODulation:DATA:Q?* to query Q data and *:MODulation:MARKer[<n>]:DATA* to query Marker data.

Return Format

The query returns a binary stream.

Examples

```
:MODulation:DATA:I? mod,1,100 /*Queries the I data of the waveform
named "mod". The data size is 100 and the start index is 1 (the
first data point).*/
```

3.8.3 :MODulation:DATA:Q?

Syntax

:MODulation:DATA:Q? <*wfm_name*>[,<*startIndex*>[,<*size*>]]

Description

Queries the Q data of the modulated waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-
<startIndex>	Integer	Refer to *Remarks*	-
<size>	Integer	Refer to *Remarks*	-

Remarks

- You can use this command to query a total number of <size> points starting from <startIndex>. startIndex+size-1 ≤ wave length. The minimum <size> is 1. If [<size>] is omitted, the length of waveform is assumed to be the value of the <size> parameter.
- You can use *:MODulation:DATA:I?* to query I data and *:MODulation:MARKer[<n>]:DATA* to query Marker data.

Return Format

The query returns a binary stream.

Examples

```
:MODulation:DATA:Q? mod,1,100 /*Queries the Q data of the waveform
named "mod". The data size is 100 and the start index is 1 (the
first data point).*/
```

3.8.4 :MODulation:MARKer[<n>]:DATA

Syntax

:MODulation:MARKer[<*n*>]:DATA <*wfm_name*>,<*startIndex*>,<*size*>,<*block_data*>

:MODulation:MARKer[<*n*>]:DATA? <*wfm_name*>[,<*startIndex*>[,<*size*>]]

Description

Sets or queries the Marker data for the specified modulated waveform.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2}	1
<wfm_name>	ASCII string	Available waveform name	-

Name	Type	Range	Default
<startIndex>	Integer	Refer to *Remarks*	-
<size>	Integer	Refer to *Remarks*	-
<block_data>	IEEE 488.2 block	Refer to *Remarks*	

Remarks

- You can use this command to set or query a total number of <size> points starting from <startIndex>. startIndex+size-1 ≤ wave length. The minimum <size> is 1.
- <block_data> is used to set the Marker data. The data is transferred in the format of TMC header+waveform data point. The TMC header is in #NXXXXX format; wherein, # is the TMC header identifier; N following # represents the length of the character string which describes the waveform length; the length of the waveform data points is expressed in ASCII strings. For example, the data read for one time is #90000000208000.... It indicates the 9 bytes are used to describe the data length. 000000020 indicates the length of waveform data, that is, 20 bytes. 8000... indicates that the first Marker is high level (80) and the second Marker is low level (00).
- You can use *:MODulation:DATA:I?* and *:MODulation:DATA:Q?* to query the I data and Q data respectively.

Return Format

The query returns a binary stream.

Examples

```
:MODulation:MARKer1:DATA mod,1,20,#9000000020xxxx... /*Edits the
Marker1 data of the modulated waveform named "mod". The data size
is 20 and the start index is 1 (the first point).*/
:MODulation:MARKer1:DATA? Mod,1,20 /*Queries the Marker1 data of
the modulated waveform named "mod". The data size is 20 and the
start index is 1.*/
```

3.8.5 :MODulation:OVERwrite

Syntax

:MODulation:OVERwrite <*state*>

:MODulation:OVERwrite?

Description

Sets or queries whether the "Overwrite existing waveform" function is enabled.

Parameter

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

Remarks

1 or ON overwrites the existing modulated waveform of the same name. 0 or OFF automatically adds the suffix "_n" (n is a positive integer) to the filename, for example, "mode_1".

Return Format

The query returns 0 or 1.

Examples

```
:MODulation:OVERwrite 1 /*Enables the "Overwrite existing waveform"
function.*/
:MODulation:OVERwrite? /*Queries whether the "Overwrite existing
waveform" function is enabled. The query returns 1.*/
```

3.8.6 :MODulation:SIGNal:CTYPe

Syntax

:MODulation:SIGNal:CTYPe <*ctype*>

:MODulation:SIGNal:CTYPe?

Description

Sets or queries the code type.

Parameter

Name	Type	Range	Default
<ctype>	Discrete	{ALL1|ALL0|PRBS9|PRBS11|PRBS15|PRBS16|PRBS20|PRBS21|PRBS23}	ALL1

Remarks

- **ALL1** create a sequence in which all bits are 1.
- **ALL0** create a sequence in which all bits are 0.
- **PRBSn** sets the type to PRBS9, PRBS11, PRBS15, PRBS16, PRBS20, PRBS21, and PRBS23.

Return Format

The query returns the code type, for example, PRBS15.

Examples

```
:MODulation:SIGNal:CTYPe PRBS15 /*Sets the code type to PRBS15.*/
:MODulation:SIGNal:CTYPe? /*Queries the code type. The query
returns PRBS15.*/
```

3.8.7 :MODulation:SIGNal:LENGth

Syntax

`:MODulation:SIGNal:LENGth` <*length*>

`:MODulation:SIGNal:LENGth?`

Description

Sets or queries the code length.

Parameter

Name	Type	Range	Default
<length>	Integer	10 to 20M	10k

Remarks

None.

Return Format

The query returns an integer, for example, 1000.

Examples

```
:MODulation:SIGNal:LENGth 1000 /*Sets the code length to 1000.*/
:MODulation:SIGNal:LENGth? /*Queries the code length. The query
returns 1000.*/
```

3.8.8 :MODulation:SIGNal:RATE

Syntax

`:MODulation:SIGNal:RATE` <*rate*>

`:MODulation:SIGNal:RATE?`

Description

Sets or queries the symbol rate in units of Sa/s.

Parameter

Name	Type	Range	Default
<rate>	Integer	100 Sa/s to 100 MSa/s	1 MSa/s

Remarks

None.

Return Format

The query returns the symbol rate in scientific notation. For example, the query might return 1.0000000000E+03, indicating the symbol rate is 1000 Sa/s.

Examples

```
:MODulation:SIGNal:RATE 1000 /*Sets the symbol rate to 1000 Sa/s.*/
:MODulation:SIGNal:RATE? /*Queries the symbol rate. The query
returns 1.0000000000E+03.*/
```

3.8.9 :MODulation:SOURce:ALPHa

Syntax

:MODulation:SOURce:ALPHa <*alpha*>

:MODulation:SOURce:ALPHa?

Description

Sets or queries the roll-off factor Alpha/BT when using filter.

Parameter

Name	Type	Range	Default
<alpha>	Integer	5 to 100	25

Remarks

- The command is valid only when the filter type (*:MODulation:SOURce:FILTer*) is set to cosine filter (COSine) or root cosine filter (ROOT).
- The <alpha> ranges from 5 to 100 and the filter roll-off factor ranges from 0.05 to 1.

Return Format

The query returns an integer. For example, the query might return 50, indicating that the roll-off factor is 0.5.

Examples

```
:MODulation:SOURce:ALPHa 50 /*Sets the roll-off factor to 0.5.*/
:MODulation:SOURce:ALPHa? /*Queries the roll-off factor. The query
returns 50.*/
```

3.8.10 :MODulation:SOURce:CTYPe

Syntax

:MODulation:SOURce:CTYPe <*ctype*>

:MODulation:SOURce:CTYPe?

Description

Sets or queries the encoding type.

Parameter

Name	Type	Range	Default
<ctype>	Discrete	{OFF|DIFF|DGRay|GRAY}	OFF

Remarks

- **OFF:** No code.
- **DIFF:** Differential coding.
- **DGRay:** Differential+Gray coding.
- **GRAY:** Gray coding.

Return Format

The query returns OFF, DIFF, DGR, or GRAY.

Examples

```
:MODulation:SOURce:CTYPe DIFF /*Sets the code type to
Differential.*/
:MODulation:SOURce:CTYPe? /*Queries the code type. The query
returns DIFF.*/
```

3.8.11 :MODulation:SOURce:FILTer

Syntax

:MODulation:SOURce:FILTer <*filter*>

:MODulation:SOURce:FILTer?

Description

Sets or queries the filter type.

Parameter

Name	Type	Range	Default
<filter>	Discrete	{RECTangular|COSine|ROOT}	RECTangular

Remarks

- **RECTangular** sets the filter to rectangular.
- **COSine** sets the filter to cosine.
- **ROOT** sets the filter to root cosine.

Return Format

The query returns RECT, COS, or ROOT.

Examples

```
:MODulation:SOURce:FILTer COSine /*Sets the filter type to cosine.*/
:MODulation:SOURce:FILTer? /*Queries the filter type. The query
returns COS.*/
```

3.8.12 :MODulation:SOURce:NAME

Syntax

:MODulation:SOURce:NAME <*name*>

:MODulation:SOURce:NAME?

Description

Sets or queries the name of the modulated waveform.

Parameter

Name	Type	Range	Default
<name>	ASCII String	The input string can be a combination of English letters, numbers, and underscores. It cannot be longer than 26 bytes.	-

Remarks

When the compile operation (*:MODulation:COMPile*) is executed, this command names the modulated waveform.

Return Format

The query returns an ASCII string, for example, Mod_2.

Examples

```
:MODulation:SOURce:NAME Mod_2 /*Sets the waveform name to Mod_2.*/
:MODulation:SOURce:NAME? /*Queries the waveform name. The query
returns Mod_2.*/
```

3.8.13 :MODulation:SOURce:OVER

Syntax

:MODulation:SOURce:OVER <*over*>

:MODulation:SOURce:OVER?

Description

Sets or queries the oversampling.

Parameter

Name	Type	Range	Default
<over>	Integer	1 to 16	4

Remarks

Oversampling refers to the interpolation and filtering of the IQ baseband modulation signal to output the signal at a higher sampling rate. If the original IQ baseband signal symbol rate Rs is raised to a signal with a sample rate of fs=R×Rs after oversampling, R is the factor by which the signal is oversampled.

Return Format

The query returns an integer, for example, 3.

Examples

```
:MODulation:SOURce:OVER 3 /*Sets the oversampling to 3.*/
:MODulation:SOURce:OVER? /*Queries the oversampling. The query
returns 3.*/
```

3.8.14 :MODulation:SOURce:TYPE

Syntax

:MODulation:SOURce:TYPE <*type*>

:MODulation:SOURce:TYPE?

Description

Sets or queries the modulation type.

Parameter

Name	Type	Range	Default
<type>	Discrete	{BPSK|QPSK|8PSK|16QAM|32QAM|64QAM|128QAM|256QAM}	BPSK

Remarks

None.

Return Format

The query returns the modulation type, for example, QPSK.

Examples

```
:MODulation:SOURce:TYPE QPSK /*Sets the modulation type to QPSK.*/
:MODulation:SOURce:TYPE? /*Queries the modulation type. The query
returns QPSK.*/
```

3.9 :OUTPut Commands

:OUTPut commands are used to set and query the channel outputs.

3.9.1 :OUTPut:OFF

Syntax

:OUTPut:OFF <*state*>

:OUTPut:OFF?

Description

Sets or queries the state (enabled or disabled) of the All Channels Off control.

Parameter

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

Remarks

None.

Return Format

The query returns 0 or 1.

Examples

```
:OUTPut:OFF 1 /*Enables All Channels Off.*/
:OUTPut:OFF? /*Queries whether all channels are off. The query
returns 1.*/
```

3.9.2 :OUTPut[<n>]:PATH

Syntax

:OUTPut[<*n*>]:PATH <*path*>

:OUTPut[<*n*>]:PATH?

Description

Sets or queries the output signal path of the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<path>	Discrete	{AC|DCHbw|DCAmp}	DCHbw

Remarks

- **AC** sets the mode to AC output mode. For outputs via the single-ended AC connector on the front panel.
- **DCHbw** sets the mode to DC high-bandwidth output mode. For outputs via the front-panel Analog DC+ and DC- output connectors.
- **DCAmp (optional)** sets the mode to DC amplifier output mode. For outputs via the front-panel Analog DC+ and DC- differential output connectors. Compared with DC HBW, DC AMP mode provides greater amplitude range.

Return Format

The query returns AC, DCA, or DCH.

Examples

```
:OUTPut1:PATH AC /*Sets the CH1 output signal path to AC.*/
:OUTPut1:PATH? /*Queries the CH1 output signal path. The query
returns AC.*/
```

3.9.3 :OUTPut[<n>][:STATe]

Syntax

:OUTPut[<*n*>][:STATe] <*state*>

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

Description

Sets or queries the output state for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

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

Remarks

- 0 or OFF disables the channel's output. 1 or ON enables the channel's output.
- [<n>] determines the channel number. If omitted, interpreted as 1.

Return Format

The query returns 0 or 1.

Examples

```
:OUTPut1:STATe 1 /*Enables the output for CH1.*/
:OUTPut1:STATe? /*Queries the output state of CH1. The query
returns 1.*/
```

3.9.4 :OUTPut[<n>]:SVALue[:ANALog][:STATe]

Syntax

:OUTPut[<*n*>]:SVALue[:ANALog][:STATe] <*state*>

:OUTPut[<*n*>]:SVALue[:ANALog][:STATe]?

Description

Sets or queries the output condition of a waveform of the specified channel while the channel is in the stopped state.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<state>	Discrete	{OFF|ZERO}	ZERO

Remarks

- OFF sets the stop state output for channel "n" to open (electrically disconnected). ZERO sets the stop state output value for channel "n" to 0 volts.
- [<n>] determines the channel number. If omitted, interpreted as 1.

Return Format

The query returns OFF or ZERO.

Examples

```
:OUTPut1:SVALue:ANALog:STATe OFF /*Sets CH1's output to be
disconnected when in the stopped state.*/
:OUTPut1:SVALue:ANALog:STATe? /*Queries CH1's output when in the
stopped state. The query returns OFF.*/
```

3.9.5 :OUTPut[<n>]:SVALue:MARKer[<m>]

Syntax

:OUTPut[<*n*>]:SVALue:MARKer[<*m*>] <*state*>

:OUTPut[<*n*>]:SVALue:MARKer[<*m*>]?

Description

Sets or queries the output condition of the specified Marker of the specified channel when in the stopped state.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<m>	Discrete	{1|2}	1
<state>	Discrete	{OFF|LOW}	LOW

Remarks

- OFF sets the stop state marker output for channel "n" to open (electrically disconnected); LOW sets the stop state marker output for channel "n" to a logic level low.
- [<n>] determines the channel number. If omitted, interpreted as 1; [<m>] determines the Marker number. If omitted, interpreted as 1.

Return Format

The query returns OFF or LOW.

Examples

```
:OUTPut1:SVALue:MARKer1 LOW /*Sets the Marker1 of CH1 to a logic
level low when in the stopped state.*/
:OUTPut1:SVALue:MARKer1? /*Queries the output condition of the
Marker1 of CH1. The query returns LOW.*/
```

3.9.6 :OUTPut[<n>]:WVALue[:ANALog][:STATe]

Syntax

:OUTPut[<*n*>]:WVALue[:ANALog][:STATe] <*state*>

:OUTPut[<*n*>]:WVALue[:ANALog][:STATe]?

Description

Sets or queries the output condition of a waveform of the specified channel while the channel is in the waiting-for-trigger state or for a brief period after the waveform loads to the DAC and before the first point plays.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<state>	Discrete	{ZERO|FIRSt}	ZERO

Remarks

- ZERO sets the output level for channel "n" to 0 volts when channel "n" is in the waiting-for-trigger state; FIRSt sets the output level for channel "n" to match the first point in the waveform when channel "n" is in the waiting-for-trigger state.
- [<n>] determines the channel number. If omitted, interpreted as 1.

Return Format

The query returns ZERO or FIRS.

Examples

```
:OUTPut1:WVALue:ANALog:STATe FIRSt /*Sets the output level for
channel 1 to match the first point in the waveform when channel 1
is in the waiting-for-trigger state.*/
:OUTPut1:WVALue:ANALog:STATe? /*Queries the output condition of a
waveform of channel 1 when channel 1 is in the waiting-for-trigger
state. The query returns FIRSt.*/
```

3.9.7 :OUTPut[<n>]:WVALue:MARKer[<m>]

Syntax

:OUTPut[<*n*>]:WVALue:MARKer[<*m*>] <*state*>

:OUTPut[<*n*>]:WVALue:MARKer[<*m*>]?

Description

Sets or queries the output condition of the specified Marker of the specified channel while the instrument is in the waiting-for-trigger state or for a brief period after the waveform loads to the DAC and before the first point plays.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<m>	Discrete	{1|2}	1
<state>	Discrete	{FIRSt|LOW|HIGH}	LOW

Remarks

- <state> specifies the output state:
  - **LOW** sets the marker output to a logic level low when the channel is in the waiting-for-trigger state.
  - **HIGH** sets the marker output to a logic level high when the channel is in the waiting-for-trigger state. The default high level is 1.00 V.
  - **FIRSt** sets the marker output level to match the first point in the marker when the channel is in the waiting-for-trigger state.
- [<n>] determines the channel number. If omitted, interpreted as 1. [<m>] determines the Marker number. If omitted, interpreted as 1.

Return Format

The query returns FIRS, LOW, or HIGH.

Examples

```
:OUTPut1:WVALue:MARKer1 FIRSt /*Sets the output level of Marker1 in
the waiting state to the value of the first point of the waveform.*/
:OUTPut1:WVALue:MARKer1? /*Queries the output level of Marker1 in
the waiting state. The query returns FIRS.*/
```

3.10 :SLISt Commands

:SLISt commands are used to set and query the specified sequence.

3.10.1 :SLISt:LIST?

Syntax

```
:SLISt:LIST?
```

Description

Queries the existing sequences in the sequence list.

Parameter

None.

Remarks

None.

Return Format

The query returns an ASCII string, for example, seq_1,seq_2,seq_3.

Examples

```
:SLISt:LIST? /*Queries the existing sequences in the sequence list.
The query might return seq_1,seq_2,seq_3.*/
```

3.10.2 :SLISt:NAME?

Syntax

:SLISt:NAME? <*seq_list_index*>

Description

Queries the name of the sequence at the specified sequence list index.

Parameter

Name	Type	Range	Default
<seq_list_index>	Integer	NR1	-

Remarks

You can send *:SLISt:SIZE?* to query the number of sequences in the sequence list.

Return Format

The query returns a string. If there is not a sequence at the chosen index, an empty string is returned.

Examples

```
:SLISt:NAME? 2 /*Queries the name of the second sequence in the
sequence list. The query returns seq_2.*/
```

3.10.3 :SLISt:SEQuence:DELete

Syntax

:SLISt:SEQuence:DELete <*seq*>

Description

Deletes a specific sequence from the sequence list.

Parameter

Name	Type	Range	Default
<seq>	ASCII string	Available sequence name	-

Remarks

None.

Return Format

None.

Examples

```
:SLISt:SEQuence:DELete Seq_1 /*Deletes the sequence named "Seq_1"
from the sequence list.*/
```

3.10.4 :SLISt:SEQuence:DELete:ALL

Syntax

`:SLISt:SEQuence:DELete:ALL`

Description

Deletes all sequences from the sequence list.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
:SLISt:SEQuence:DELete:ALL /*Deletes all sequences from the
sequence list.*/
```

3.10.5 :SLISt:SEQuence:EVENt:JTIMing

Syntax

`:SLISt:SEQuence:EVENt:JTIMing` <*seq_name*>,<*type*>

`:SLISt:SEQuence:EVENt:JTIMing?` <*seq_name*>

Description

Sets or queries when an event jump or pattern jump in the sequence occurs (Jump Timing).

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-
<type>	Discrete	{END|IMMediate}	IMMediate

Remarks

- **END:** When a trigger signal is received, a jump occurs only after the currently playing waveform completes its playout.
- **IMMediate:** A jump occurs immediately at the time a trigger signal is received.

Return Format

The query returns END or IMM.

Examples

```
:SLISt:SEQuence:EVENt:JTIMing Seq_1,END /*Sets the Jump Timing to
Jump At End for the sequence named "Seq_1".*/
:SLISt:SEQuence:EVENt:JTIMing? Seq_1 /*Queries the Jump Timing for
the sequence named "Seq_1". The query returns END.*/
```

3.10.6 :SLISt:SEQuence:EVENt:PJUMp:DEFine

Syntax

:SLISt:SEQuence:EVENt:PJUMp:DEFine <*seq_name*>,<*pattern*>,<*jump_step*>

:SLISt:SEQuence:EVENt:PJUMp:DEFine? <*seq_name*>,<*pattern*>

Description

Sets or queries the jump step associated to the specified pattern for the specified sequence.

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-
<pattern>	Integer	0 to 255	-
<jump_step>	Integer	1 to 16384	-

Remarks

DG70000 series allows you to send 8 bit command via the **[Pattern Jump In]** connector. When the instrument receives the command, it performs 8 bit addressing and jump to the specified step based on predefined jump destinations. Up to 256 predefined input patterns are available for DG70000. The digital pattern is applied to the **[Pattern Jump In]** connector on the rear panel.

- <pattern> specifies the input pattern to make a pattern jump, expressed as a decimal integer.
- <jump_step> defines the target step to jump to for the specified pattern.

Return Format

The query returns an integer, for example, 8.

Examples

```
:SLISt:SEQuence:EVENt:PJUMp:DEFine Seq_1,15,3 /*Sets the target
jump step to the step 3 of "Seq_1" for the event pattern 00001111.*/
:SLISt:SEQuence:EVENt:PJUMp:DEFine? Seq_1,15 /*Queries the target
jump step of "Seq_1" for the event pattern 00001111. The query
returns 3.*/
```

3.10.7 :SLISt:SEQuence:EVENt:PJUMp:ENABle

Syntax

:SLISt:SEQuence:EVENt:PJUMp:ENABle <*seq_name*>,<*state*>

:SLISt:SEQuence:EVENt:PJUMp:ENABle? <*seq_name*>

Description

Sets or queries the Pattern Jump on/off state for the specified sequence.

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-
<state>	Bool	{0|1|OFF|ON}	1

Remarks

When enabled, the data at the **[Pattern Jump In]** connector can be strobed in, causing a sequence to jump to a specified step. The sequence and step are defined with the command *:SLISt:SEQuence:EVENt:PJUMp:DEFine*.

Return Format

The query returns 0 or 1.

Examples

```
:SLISt:SEQuence:EVENt:PJUMp:ENABle Seq_1,OFF /*Disables the pattern
jump for "Seq_1".*/
:SLISt:SEQuence:EVENt:PJUMp:ENABle? Seq_1 /*Queries whether the
pattern jump is enabled for "Seq_1". The query returns 0.*/
```

3.10.8 :SLISt:SEQuence:EVENt:PJUMp:SIZE?

Syntax

:SLISt:SEQuence:EVENt:PJUMp:SIZE?

Description

Queries the maximum number of entries in the pattern jump table.

Parameter

None.

Remarks

None.

Return Format

The query returns an integer, 256.

Examples

```
:SLISt:SEQuence:EVENt:PJUMp:SIZE? /*Queries the maximum number of
entries in the pattern jump table. The query returns 256.*/
```

3.10.9 :SLISt:SEQuence:LENGth?

Syntax

:SLISt:SEQuence:LENGth? <*seq_name*>

Description

Queries the total number of steps in the specified sequence.

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-

Remarks

None.

Return Format

The query returns an integer.

Examples

```
:SLISt:SEQuence:LENGth? Seq_1 /*Queries the total number of steps
in "Seq_1". The query might return 100.*/
```

3.10.10 :SLISt:SEQuence:NEW

Syntax

:SLISt:SEQuence:NEW <*seq_name*>[,<*number_of_tracks*>]

Description

Creates a new sequence with the selected name and number of tracks.

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-
<number_of_tracks>	Integer	1 to 4	1

Remarks

- You are not allowed to create a new sequence of an existing sequence name. You can use *:SLISt:LIST?* to acquire all existing sequence names.
- When [<number_of_tracks>] is omitted, the created sequence contains one Track.

Return Format

None.

Examples

```
:SLISt:SEQuence:NEW Seq,2 /*Creates a new sequence named "Seq" with
two tracks.*/
```

3.10.11 :SLISt:SEQuence:QUICk

Syntax

:SLISt:SEQuence:QUICk <*seq*>

Description

Selects the specified sequence for quick save mode.

Parameter

Name	Type	Range	Default
<seq>	ASCII string	Available sequence name	-

Remarks

After using this command to select a sequence, you can use *:SLISt:SEQuence:QUICk:STATe* to enable the quick save mode for the selected sequence.

Return Format

None.

Examples

```
:SLISt:SEQuence:QUICk seq/*Selects the sequence named "seq".*/
```

3.10.12 :SLISt:SEQuence:QUICk:STATe

Syntax

:SLISt:SEQuence:QUICk:STATe <*bool*>

Description

Enables or disables the quick save mode.

Parameter

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

Remarks

- Please first use *:SLISt:SEQuence:QUICk* to select the specified sequence.
- 1 or ON enables the sequence quick save mode. In this mode, the modifications on the specified sequence will not be updated in real time. When this mode is disabled, all modifications will be batch saved at a time. The quick save function can help improve the efficiency of sequence editing.
- You can select only one sequence to use the quick save function.
- If this mode is not disabled for a specified sequence before another sequence is selected (*:SLISt:SEQuence:QUICk*), all modifications on the sequence will be lost.

Return Format

None.

Examples

```
None.
```

3.10.13 :SLISt:SEQuence:STEP:MAX?

Syntax

:SLISt:SEQuence:STEP:MAX?

Description

Queries the maximum number of steps allowed in a sequence.

Parameter

None.

Remarks

None.

Return Format

The query returns an integer.

Examples

```
:SLISt:SEQuence:STEP:MAX? /*Queries the maximum number of steps
allowed in a sequence. The query returns 16384.*/
```

3.10.14 :SLISt:SEQuence:STEP:RCOunt:MAX?

Syntax

:SLISt:SEQuence:STEP:RCOunt:MAX?

Description

Queries the maximum number of repeats allowed for a step in a sequence.

Parameter

None.

Remarks

None.

Return Format

The query returns an integer.

Examples

```
:SLISt:SEQuence:STEP:RCOunt:MAX? /*Queries the maximum number of
repeats allowed for a step in a sequence. The query might return
4294967295.*/
```

3.10.15 :SLISt:SEQuence:STEP[<n>]:EJINput

Syntax

:SLISt:SEQuence:STEP[<*n*>]:EJINput <*seq_name*>,<*type*>

:SLISt:SEQuence:STEP[<*n*>]:EJINput? <*seq_name*>

Description

Sets or queries whether the specified sequence will jump when it receives Trigger A, Trigger B, Internal Trigger, or no jump at all.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<seq_name>	ASCII string	Available sequence name	-
<type>	Discrete	{ATRigger|BTRigger|OFF|ITRigger}	OFF

Remarks

This command defines whether an event jump is active and the type of trigger signal for the jump to occur. If a trigger signal is received, the sequence uses the Event Jump To definition to jump to a specified step.

- **OFF:** A jump is not active for the step. The sequence uses the Go To definition after finishing this step (*:SLISt:SEQuence:STEP[<n>]:GOTO*).
- **ATRigger:** A jump occurs when a trigger signal Trig A is received.
- **BTRigger:** A jump occurs when a trigger signal Trig B is received.
- **ITRigger:** A jump occurs when an internal trigger signal is received.

Return Format

The query returns ATR, BTR, OFF, or ITR.

Examples

```
:SLISt:SEQuence:STEP1:EJINput Seq_1,ATRigger /*Sets the trigger
source to TrigA for the step 1 of the sequence named "Seq_1".*/
:SLISt:SEQuence:STEP1:EJINput? Seq_1 /*Queries the trigger source
for the step 1 of the sequence named "Seq_1". The query returns
ATR.*/
```

3.10.16 :SLISt:SEQuence:STEP[<n>]:EJUMp

Syntax

:SLISt:SEQuence:STEP[<*n*>]:EJUMp <*seq_name*>,{<*entry*>|*NEXT*|*FIRSt*|*LAST*}

:SLISt:SEQuence:STEP[<*n*>]:EJUMp? <*seq_name*>

Description

Sets or queries the step that the specified sequence will jump to on a trigger event.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<seq_name>	ASCII string	Available sequence name	-
<entry>	Integer	1 to 16384	-

Remarks

- This command sets the step that the specified sequence will jump to when a trigger signal is received. You can select the NEXT, FIRSt, or LAST step to jump to. You can also use <entry> to self-define the step number.
- This command is valid only when an event jump (*:SLISt:SEQuence:STEP[<n>]:EJINput*) is active for the specified sequence step.

Return Format

The query returns NEXT, FIRS, LAST, or an integer.

Examples

```
:SLISt:SEQuence:STEP1:EJUMp Seq_1,2 /*Enables the sequence named
"Seq_1" to jump to step 2 after executing the step 1 on a trigger
event.*/
:SLISt:SEQuence:STEP1:EJUMp? Seq_1 /*Queries the target step that
the "Seq_1" will jump to after executing the step 1 on a trigger
event. The query returns 2.*/
```

3.10.17 :SLISt:SEQuence:STEP[<n>]:GOTO

Syntax

:SLISt:SEQuence:STEP[<*n*>]:GOTO <*seq_name*>,{<*entry*>|*NEXT*|*FIRSt*|*LAST*|*END*}

:SLISt:SEQuence:STEP[<*n*>]:GOTO? <*seq_name*>

Description

Sets or queries the target step for the GO TO jump of the sequence at the specified step.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<seq_name>	ASCII string	Available sequence name	-
<entry>	Integer	1 to 16384	-

Remarks

The commands defines which step in the sequence to jump to and play when the current step has finished playing its waveform. The default is NEXT.

- **<entry>** allows you to define the specified step.
- **NEXT** enables the sequence to go to the next step.
- **FIRSt** enables the sequence to go to the first step.
- **LAST** enables the sequence to go to the last step.
- **END** enables the sequence to go to the end. If the subsequence uses a step definition of End (end of sequence), it goes to the main sequence and continues to play.

Return Format

The query returns NEXT, FIRS, LAST, END, or an integer.

Examples

```
:SLISt:SEQuence:STEP1:GOTO Seq_1,2 /*Enables the sequence named
"Seq_1" to jump to step 2 after executing step 1.*/
:SLISt:SEQuence:STEP1:GOTO? Seq_1 /*Queries the target step to go
to for the step 1 of "Seq_1". The query returns 2.*/
```

3.10.18 :SLISt:SEQuence:STEP[<n>]:RCOunt

Syntax

:SLISt:SEQuence:STEP[<*n*>]:RCOunt <*seq_name*>,{<*cnt*>|*ONCE*|*INFinite*}

:SLISt:SEQuence:STEP[<*n*>]:RCOunt? <*seq_name*>

Description

Sets or queries the repeat count for the specified step of the named sequence.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<seq_name>	ASCII string	Available sequence name	-
<cnt>	Integer	1 to 4294967295	1

Remarks

- **ONCE** plays the waveform one time during the specified sequence step.
- **INFinite** sets the repeat count to infinity. The sequence will continuously play the waveform until a jump condition occurs.
- **<cnt>** specifies the number of times to play the waveform during this sequence step. The allowed value is between 1 and 4294967295 (2^{32}-1).

Return Format

The query returns ONCE, INF, or an integer.

Examples

```
:SLISt:SEQuence:STEP1:RCOunt Seq_1,2/*Sets the repeat count to 2
for step 1 of the sequence named "Seq_1".*/
:SLISt:SEQuence:STEP1:RCOunt? Seq_1 /*Queries the repeat count for
step 1 of the sequence named "Seq_1". The query returns 2.*/
```

3.10.19 :SLISt:SEQuence:STEP[<n>]:TASSet[<m>]?

Syntax

:SLISt:SEQuence:STEP[<*n*>]:TASSet[<*m*>]? <*seq_name*>

Description

Queries the name of the waveform or subsequence at the specified sequence's step and track.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<m>	Integer	1 to 4	1
<seq_name>	ASCII string	Available sequence name	-

Remarks

- [<n>] specifies the step number and [<m>] specifies the Track number.
- You can send *:SLISt:SEQuence:STEP[<n>]:TASSet[<m>]:TYPE?* to query the asset type (waveform or subsequence) at the specified location.

Return Format

The query returns an ASCII string. An empty string is returned if no waveform or sequence has been assigned to the track and step.

Examples

```
:SLISt:SEQuence:STEP1:TASSet1? Seq_1 /*Queries the name of the
asset at the first step of Track1 for sequence named "Seq_1". The
query might return wave.*/
```

3.10.20 :SLISt:SEQuence:STEP[<n>]:TASSet:SEQuence

Syntax

:SLISt:SEQuence:STEP[<*n*>]:TASSet:SEQuence <*seq_name*>,<*subseq_name*>

Description

Assigns a subsequence for a specific sequence's step of all Tracks.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<seq_name>	ASCII string	Available sequence name	-
<subseq_name>	ASCII string	Available sequence name	-

Remarks

You can send *:SLISt:SEQuence:STEP[<n>]:TASSet[<m>]?* to query the name of the waveform or subsequence assigned to a specific sequence's step and track.

Return Format

None.

Examples

```
:SLISt:SEQuence:STEP1:TASSet:SEQuence Seq_1,Seq_sub /*Adds the
subsequence named "Seq_sub" to the step 1 of all tracks in the
sequence named "Seq_1".*/
```

3.10.21 :SLISt:SEQuence:STEP[<n>]:TASSet[<m>]:TYPE?

Syntax

:SLISt:SEQuence:STEP[<*n*>]:TASSet[<*m*>]:TYPE? <*seq_name*>

Description

Queries the type of asset (waveform or subsequence) assigned to the step and track for a specified sequence.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<m>	Integer	1 to 4	1
<seq_name>	ASCII string	Available sequence name	-

Remarks

[<n>] specifies the step number and [<m>] specifies the Track number.

Return Format

The query returns WAV (waveform) or SEQ (subsequence).

Examples

```
:SLISt:SEQuence:STEP1:TASSet1:TYPE? Seq_1 /*Queries the type of
asset in the first step of Track1 for the sequence named "Seq_1".
The query might return WAV.*/
```

3.10.22 :SLISt:SEQuence:STEP[<n>]:TASSet[<m>]:WAVeform

Syntax

:SLISt:SEQuence:STEP[<*n*>]:TASSet[<*m*>]:WAVeform
<*seq_name*>,<*waveform_name*>

Description

Assigns a waveform for a specific sequence's step and track.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<m>	Integer	1 to 4	1

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-
<waveform_name>	ASCII string	Available waveform name	-

Remarks

[<n>] specifies the step number and [<m>] specifies the Track number.

Return Format

None.

Examples

```
:SLISt:SEQuence:STEP1:TASSet1:WAVeform Seq_1,Wave_1 /*Assigns the
waveform named "Wave_1" to the step 1 of Track1 for the sequence
named "Seq_1".*/
```

3.10.23 :SLISt:SEQuence:STEP[<n>]:WINPut

Syntax

:SLISt:SEQuence:STEP[<*n*>]:WINPut <*seq_name*>,<*type*>

:SLISt:SEQuence:STEP[<*n*>]:WINPut? <*seq_name*>

Description

Sets or queries the wait conditions for the specified step in a sequence, including whether to wait for the trigger source for a step to play and the type of trigger source.

Parameter

Name	Type	Range	Default
<n>	Integer	1 to 16384	1
<seq_name>	ASCII string	Available sequence name	-
<type>	Discrete	{ATRigger|BTRigger|OFF|ITRigger}	OFF

Remarks

- **ATRigger:** The specified step does not start playing the waveform until a trigger signal TrigA is received.
- **BTRigger:** The specified step does not start playing the waveform until a trigger signal TrigB is received.
- **OFF:** No waiting. The waveform plays immediately without waiting for trigger.

- **ITRigger:** The step does not start playing the waveform until an internal trigger signal is received.

Return Format

The query returns ATR, BTR, OFF, or ITR.

Examples

```
:SLISt:SEQuence:STEP1:WINPut Seq,ATRigger /*Sets the step 1 of
"Seq" to wait for the trigger signal TrigA.*/
:SLISt:SEQuence:STEP1:WINPut? Seq /*Queries the wait conditions for
the step 1 of "Seq". The query returns ATR.*/
```

3.10.24 :SLISt:SEQuence:TRACk?

Syntax

:SLISt:SEQuence:TRACk? <*seq_name*>

Description

Queries the number of tracks defined in the specified sequence.

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-

Remarks

None.

Return Format

The query return an integer, for example, 2.

Examples

```
:SLISt:SEQuence:TRACk? Seq /*Queries the number of tracks defined
in the sequence named "Seq". The query returns 2.*/
```

3.10.25 :SLISt:SEQuence:TRACk:MAX?

Syntax

:SLISt:SEQuence:TRACk:MAX?

Description

Queries the maximum number of tracks allowed in a sequence.

Parameter

None.

Remarks

None.

Return Format

The query returns an integer, for example, 4.

Examples

```
:SLISt:SEQuence:TRACk:MAX? /*Queries the maximum number of tracks
allowed in a sequence. The query returns 4.*/
```

3.10.26 :SLISt:SEQuence:TSTamp?

Syntax

:SLISt:SEQuence:TSTamp? <*seq_name*>

Description

Queries the timestamp of the named sequence.

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-

Remarks

Queries the timestamp that indicates when the sequence was created or last modified.

Return Format

The query returns the sequence timestamp in string with "yyyy-ll-dd hh:mm:ss". Where:

- yyyy refers to a four-digit year number.
- ll refers to two-digit month number from 01 to 12.
- dd refers to two-digit day number in the month.
- hh refers to two-digit hour number.
- mm refers to two-digit minute number.
- ss refers to two-digit second number.

Examples

```
:SLISt:SEQuence:TSTamp? Seq /*Queries the timestamp of the sequence
named "Seq". The query returns the date and time the sequence named
"Seq" was created or last modified. For example, the query might
return 2022-01-18 09:11:37.*/
```

3.10.27 :SLISt:SEQuence:WMUSage?

Syntax

`:SLISt:SEQuence:WMUSage?` *<seq_name>*,*<track_number>*

Description

Queries the total waveform memory usage (in sample points) for the specified sequence track for the named sequence.

Parameter

Name	Type	Range	Default
<seq_name>	ASCII string	Available sequence name	-
<track_number>	Integer	1 to 4	-

Remarks

None.

Return Format

The query returns an integer.

Examples

```
:SLISt:SEQuence:WMUSage? Seq,1 /*Queries the total waveform memory
usage for Track1 in the sequence named "Seq". The query might
return 12000, indicating that the total waveform memory used by
Track1 is 12k.*/
```

3.10.28 :SLISt:SIZE?

Syntax

`:SLISt:SIZE?`

Description

Queries the number of sequences in sequence list.

Parameter

None.

Remarks

You can use *:SLISt:LIST?* to obtain the names of all sequences in the sequence list.

Return Format

The query returns an integer, for example, 15.

Examples

```
:SLISt:SIZE? /*Queries the total number of sequences in sequence
list. The query might return 15.*/
```

3.11 :SOURce Commands

[:SOURce] commands are used to set and query channel parameters.

3.11.1 [:SOURce]:FREQuency[:CW][:FIXed]

Syntax

[:SOURce]:FREQuency[:CW][:FIXed] <*frequency*>

[:SOURce]:FREQuency[:CW][:FIXed]?

Description

Sets or queries the sample rate in AWG mode.

Parameter

Name	Type	Range	Default
<frequency>	Real	100 Sa/s to 5 GSa/s	5 GSa/s

Remarks

The command is not valid when the clock source (*:CLOCk:SOURce*) is set to external sample clock (EXTernal).

Return Format

The query returns the AWG sample rate in scientific notation. For example, the query might return 1.0000000000E+09, indicating the sample rate is 1 GSa/s.

Examples

```
:SOURce:FREQuency:CW:FIXed 1000000000 /*Sets the sample rate in AWG
mode to 1 GSa/s.*/
:SOURce:FREQuency:CW:FIXed? /*Queries the sample rate in AWG mode.
The query returns 1.0000000000E+09.*/
```

3.11.2 [:SOURce]:SYNC

Syntax

[:SOURce]:SYNC <*val1*>[,<*val2*>[,<*val3*>[,<*val4*>]]]

[:SOURce]:SYNC?

Description

Sets or queries the synced channels.

Parameter

Name	Type	Range	Default
<val1>	Discrete	{CH1|CH2|CH3|CH4|NONE|ALL}	-
<val2>	Discrete	{CH1|CH2|CH3|CH4}	-
<val3>	Discrete	{CH1|CH2|CH3|CH4}	-
<val4>	Discrete	{CH1|CH2|CH3|CH4}	-

Remarks

You can synchronize 2 to 4 channels from CH1 to CH4. For synchronized channels, their channel parameters, run mode, and trigger source change simultaneously. Put any one of those channels in Stopped/Playing state and all synced channels will keep playing or stop playing simultaneously.

Return Format

The query returns a string, for example, CH1,CH2.

Examples

```
:SOURce:SYNC CH1,CH2 /*Synchronizes CH1 and CH2.*/
:SOURce:SYNC? /*Queries the synced channels. The query returns
CH1,CH2.*/
```

3.11.3 [:SOURce]:SYNC:STATe?

Syntax

[:SOURce]:SYNC:STATe?

Description

Queries whether the channel synchronization has been complete.

Parameter

None.

Remarks

None.

Return Format

The query returns WAIT, SUCC, or FAIL.

Examples

```
:SOURce:SYNC:STATe? /*Queries whether the channel synchronization
has been complete.*/
```

3.11.4 [:SOURce[<n>]]:CASSet?

Syntax

```
[:SOURce[<n>]]:CASSet?
```

Description

Queries the asset name for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns a string. If there is no asset assigned to the specified channel, the query returns a null string.

Examples

```
:SOURce1:CASSet? /*Queries the asset name for CH1.*/
```

3.11.5 [:SOURce[<n>]]:CASSet:CLEar

Syntax

```
[:SOURce[<n>]]:CASSet:CLEar
```

Description

Clears the assets in the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

None.

Examples

```
:SOURce1:CASSet:CLEar /*Clears the assets in CH1.*/
```

3.11.6 [:SOURce[<n>]]:CASSet:MODulation

Syntax

[:SOURce[<*n*>]]:CASSet:MODulation <*mod_name*>

Description

Assigns a modulated waveform (from the modulation list) to the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<mod_name>	ASCII string	Existing modulated waveform name	-

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

None.

Example

```
:SOURce1:CASSet:MODulation mode /*Assigns the modulated waveform
named "mode" to CH1.*/
```

3.11.7 [:SOURce[<n>]]:CASSet:SEQuence

Syntax

[:SOURce[<*n*>]]:CASSet:SEQuence <*seq_name*>,<*track_number*>

Description

Assigns a Track of a sequence to the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<seq_name>	ASCII string	Existing sequence name	-

Name	Type	Range	Default
<track_number>	Integer	1 to 4	-

Remarks

If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

None.

Examples

```
:SOURce1:CASSet:SEQuence Seq,1 /*Assigns Track1 of "Seq" to CH1.*/
```

3.11.8 [:SOURce[<n>]]:CASSet:TYPE?

Syntax

[:SOURce[<*n*>]]:CASSet:TYPE?

Description

Queries the type of the asset (waveform, sequence, or modulation) assigned to a channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns WAV (waveform), SEQ (sequence), MOD (modulation), or NONE (no asset).

Examples

```
:SOURce1:CASSet:TYPE? /*Queries the type of the asset assigned to
CH1. The query might return SEQ.*/
```

3.11.9 [:SOURce[<n>]]:CASSet:WAVeform

Syntax

[:SOURce[<*n*>]]:CASSet:WAVeform <*wfm_name*>

Description

Assigns a waveform (from the waveform list) to the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<wfm_name>	ASCII string	Existing waveform name	-

Remarks

If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

None.

Examples

```
:SOURce1:CASSet:WAVeform Wave /*Assigns the waveform named "Wave"
to CH1.*/
```

3.11.10 [:SOURce[<n>]]:CFRequency

Syntax

[:SOURce[<*n*>]]:CFRequency <*center_frequency*>

[:SOURce[<*n*>]]:CFRequency?

Description

Sets or queries the center frequency of the IQ wave in the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<center_frequency>	Real	DC to 5 GHz	0 Hz

Remarks

- This command is valid only when the selected channel is set to output IQ waveforms (*[:SOURce[<n>]]:WMODe*).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 4.0000000000E+09, indicating that the center frequency is 4 GHz.

Examples

```
:SOURce1:CFRequency 4000000000 /*Sets the center frequency to 4 GHz
for CH1.*/
:SOURce1:CFRequency?/*Queries the center frequency for CH1. The
query returns 4.0000000000E+09.*/
```

3.11.11 [:SOURce[<n>]]:DAC:RESolution

Syntax

[:SOURce[<*n*>]]:DAC:RESolution <*resolution*>

[:SOURce[<*n*>]]:DAC:RESolution?

Description

Sets or queries the DAC resolution for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<resolution>	Discrete	{14|15|16}	16

Remarks

If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

<resolution> has the following options:

- **16** represents 16+0Marker, indicating 16 bits for the output waveform with no bit for marker.
- **15** represents 15+1Marker, indicating 15 bits for the output waveform and 1 bit for marker-Marker 1.
- **14** represents 14+2Marker, indicating 14 bits for the output waveform and 2 bits for markers-Marker1 and Marker2.

Return Format

The query returns 14, 15, or 16.

Examples

```
:SOURce1:DAC:RESolution 14 /*Sets 14 bits for the output waveform
and 2 bits for markers for CH1.*/
:SOURce1:DAC:RESolution? /*Queries the DAC resolution for CH1. The
query returns 14.*/
```

3.11.12 [:SOURce[<n>]]:DDR

Syntax

[:SOURce[<*n*>]]:DDR <*mode*>

[:SOURce[<*n*>]]:DDR?

Description

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

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<mode>	Bool	{0|1|ON|OFF}	0

Remarks

- If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- When the DDR mode is enabled, the DAC mode (*[:SOURce[<n>]]:DMODe*) is automatically switched to NRZ.

Return Format

The query returns 0 or 1.

Examples

```
:SOURce1:DDR 1 /*Enables DDR mode for CH1.*/
:SOURce1:DDR? /*Queries the DDR on/off state for CH1. The query
returns 1.*/
```

3.11.13 [:SOURce[<n>]]:DMODe

Syntax

[:SOURce[<*n*>]]:DMODe <*mode*>

[:SOURce[<*n*>]]:DMODe?

Description

Sets or queries the DAC mode of the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Name	Type	Range	Default
<mode>	Discrete	{NRZ|MIX|RZ}	NRZ

Remarks

Available DAC modes include the following:

- **NRZ:** non-return-to-zero line code, in which low voltage level represents logic state 0 and high voltage level represents logic state 1.
- **RZ:** return-to-zero line code which has certain characteristics. During a cycle, bits of data are transmitted in binary states and the signal returns to zero after the data bit pulse. The falling edge represents 0 and the rising edge represents 1. It is not available in DDR mode (*[:SOURce[<n>]]:DDR*).
- **MIX:** mixed mode. The falling edge sample is just a supplement to the rising edge sample. It is not available in DDR mode (*[:SOURce[<n>]]:DDR*).

If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns NRZ, MIX, or RZ.

Examples

```
:SOURce1:DMODe RZ /*Sets the CH1 DAC mode to RZ.*/
:SOURce1:DMODe? /*Queries the CH1 DAC mode. The query returns RZ.*/
```

3.11.14 [:SOURce[<n>]]:IQIMode

Syntax

[:SOURce[<*n*>]]:IQIMode <*mode*>

[:SOURce[<*n*>]]:IQIMode?

Description

Sets or queries the interpolation in IQ mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<mode>	Discrete	{I2X|I4X|I8X}	-

Remarks

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

- This command is valid only when the selected channel is set to output IQ waveforms (*[:SOURce[<n>]]:WMODe*).
- When the clock frequency ranges from 2.5 GHz to 5 GHz, 2× and 4× are available; when the clock frequency exceeds 5 GHz (only available in IQ mode), only 8× is available.

Return Format

The query returns I2X, I4X, or I8X.

Examples

```
:SOURce1:IQIMode I2X /*Sets the interpolation to 2X in IQ mode for
CH1.*/
:SOURce1:IQIMode? /*Queries the interpolation in IQ mode for CH1.
The query returns I2X.*/
```

3.11.15 [:SOURce[<n>]]:IQPath

Syntax

[:SOURce[<*n*>]]:IQPath <*path*>

[:SOURce[<*n*>]]:IQPath?

Description

Sets or queries the IQ modulation mode (internal or external).

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<path>	Discrete	{INT|EXT}	INT

Remarks

- **INT** sets the modulation mode to internal. The I/Q signals are generated inside the instrument.
- **EXT** sets the modulation mode to external. The I/Q signals are input signals from **[MOD IN2]** interface on the rear panel.

This command is valid only when the selected channel is set to output IQ waveforms (*[:SOURce[<n>]]:WMODe*).

Return Format

The query returns INT or EXT.

Examples

```
:SOURce1:IQPath EXT /*Sets the IQ modulation mode to external.*/
:SOURce1:IQPath? /*Queries the IQ modulation mode. The query
returns EXT.*/
```

3.11.16 [:SOURce[<n>]]:JUMP:FORCe

Syntax

[:SOURce[<*n*>]]:JUMP:FORCe {<*entry*>|*FIRSt*|*CURRent*|*LAST*|*END*}

Description

Forces the sequence to jump to a specific step for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<entry>	Integer	1 to 16384	-

Remarks

- Selecting different parameters enables the sequence of the specified channel to jump to different steps. Options include FIRSt, CURRent, LAST, and END. You can also use <entry> to self-define the step number.
- Jump timing is decided by *:SLISt:SEQuence:EVENt:JTIMing*.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

None.

Examples

```
:SOURce1:JUMP:FORCe 2 /*Forces the sequence in CH1 to jump to step
2.*/
```

3.11.17 [:SOURce[<n>]]:MARKer[<m>]:DELay

Syntax

[:SOURce[<*n*>]]:MARKer[<*m*>]:DELay <*delay*>

[:SOURce[<*n*>]]:MARKer[<*m*>]:DELay?

Description

Sets or queries the delay of the specified marker for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<m>	Discrete	{1|2}	1
<delay>	Real	-2.000 ns to 2.000 ns	0 ps

Remarks

- [<n>] determines the channel number. When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- [<m>] determines the marker number. If omitted, interpreted as 1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-9, indicating that marker delay is set to 1 ns.

Examples

```
:SOURce1:MARKer1:DELay 0.000000001 /*Sets the Marker1 delay to 1 ns
for CH1.*/
:SOURce1:MARKer1:DELay? /*Queries the Marker1 delay for CH1. The
query returns 1.0000000000E-9.*/
```

3.11.18 [:SOURce[<n>]]:MARKer[<m>]:VOLTage[:LEVel][:IMMediate][:AMPLitude]

Syntax

[:SOURce[<*n*>]]:MARKer[<*m*>]:VOLTage[:LEVel][:IMMediate][:AMPLitude] <*amp*>

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

Description

Sets or queries the amplitude of the specified marker for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<m>	Discrete	{1|2}	1
<amp>	Real	200 mV to 1.75 V	1 V

Remarks

- [<n>] determines the channel number. When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- [<m>] determines the marker number. If omitted, interpreted as 1.

Return Format

The query returns the value in scientific notation. The unit is V. For example, the query might return 1.5000000000E+00.

Examples

```
:SOURce1:MARKer1:VOLTage:LEVel:IMMediate:AMPLitude 1.5 /*Sets the
amplitude of Marker1 to 1.5 V for CH1.*/
:SOURce1:MARKer1:VOLTage:LEVel:IMMediate:AMPLitude? /*Queries the
amplitude of Marker1 for CH1. The query returns 1.5000000000E+00.*/
```

3.11.19 [:SOURce[<n>]]:MARKer[<m>]:VOLTage[:LEVel][:IMMediate]:HIGH

Syntax

[:SOURce[<*n*>]]:MARKer[<*m*>]:VOLTage[:LEVel][:IMMediate]:HIGH <*high*>

[:SOURce[<*n*>]]:MARKer[<*m*>]:VOLTage[:LEVel][:IMMediate]:HIGH?

Description

Sets or queries the high level of the specified marker for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<m>	Discrete	{1|2}	1
<high>	Real	200 mV to 1.75 V	1 V

Remarks

- [<n>] determines the channel number. When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- [<m>] determines the marker number. If omitted, interpreted as 1.

Return Format

The query returns the value in scientific notation. The unit is V. For example, the query might return 1.5000000000E+00.

Examples

```
:SOURce1:MARKer1:VOLTage:LEVel:IMMediate:HIGH 1.5 /*Sets the high
level of Marker1 to 1.5 V for CH1.*/
:SOURce1:MARKer1:VOLTage:LEVel:IMMediate:HIGH? /*Queries the high
level of Marker1 for CH1. The query returns 1.5000000000E+00.*/
```

3.11.20 [:SOURce[<n>]]:POWer[:LEVel][:IMMediate][:AMPLitude]

Syntax

[:SOURce[<*n*>]]:POWer[:LEVel][:IMMediate][:AMPLitude] <*amp*>

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

Description

Sets or queries the amplitude for the waveform associated with the specified channel in units of dBm.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<amp>	Real	Refer to *Remarks*	Refer to *Remarks*

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

The amplitude range and default value depend on the selected output path (*:OUTPut[<n>]:PATH*). The unit is dBm.

- In **DC HBW**, the amplitude ranges from -5.14 dBm to +0.88 dBm, and the default is -5.14 dBm.
- In **DC AMP (optional)**, the amplitude ranges from -28.6 dBm to +3.98 dBm, and the default is -28.6 dBm.
- In **AC**, the amplitude ranges from -22.04 dBm to +10 dBm, and the default is -22.04 dBm.

Return Format

The query returns the value in scientific notation. For example, the query might return -6.0000000E-01, indicating that the amplitude is -0.6 dBm.

Examples

```
:SOURce1:POWer:LEVel:IMMediate:AMPLitude -0.6 /*Sets the amplitude
to -0.6 dBm for CH1.*/
:SOURce1:POWer:LEVel:IMMediate:AMPLitude? /*Queries the amplitude
of CH1. The query returns -6.0000000E-01.*/
```

3.11.21 [:SOURce[<n>]]:RMODe

Syntax

[:SOURce[<*n*>]]:RMODe <*mode*>

[:SOURce[<*n*>]]:RMODe?

Description

Sets or queries the run mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<mode>	Discrete	{CONTinuous|TRIGgered|TCONtinuous|GATed}	CONTinuous

Remarks

- **CONTinuous** sets the run mode to Continuous. Waveform starts to play and continues to repeat, without the need for a trigger event.
- **TRIGgered** sets the run mode to Triggered. Waveform starts to play and stops after one complete waveform cycle when a trigger event occurs. Waveform playout cannot be retriggered until the current waveform playout completes an entire cycle.
- **TCONtinuous** sets the run mode to Trig'd Cont. Waveform starts to play continuously until stopped by the user once a trigger event occurs. Retriggering is not required in waveform playout.
- **GATed** sets the run mode to Gated. Waveform starts to play when it is within the high level threshold of the trigger signal.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns CONT, TRIG, TCON or GAT.

Examples

```
:SOURce1:RMODe TRIGgered /*Sets the run mode to Triggered for CH1.*/
:SOURce1:RMODe? /*Queries the run mode of CH1. The query returns
TRIG.*/
```

3.11.22 [:SOURce[<n>]]:SCSTep?

Syntax

[:SOURce[<*n*>]]:SCSTep?

Description

Queries the current step of the sequence for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

[<n>] determines the channel number. When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns a value between 1 and 16384.

Examples

```
:SOURce1:SCSTep? /*Queries the current step of the sequence in CH1.
The query might return 15, indicating that CH1 is currently at step
15 of the sequence.*/
```

3.11.23 [:SOURce[<n>]]:SIFI:MODE

Syntax

[:SOURce[<*n*>]]:SIFI:MODE <*mode*>

[:SOURce[<*n*>]]:SIFI:MODE?

Description

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

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<mode>	Discrete	{SIFI|DIRect}	DIR

Remarks

- "SIFI" enables the SiFi function; "DIRect" disables the SiFi function.

- This command is valid only when the selected channel is set to output real waveforms (*[:SOURce[<n>]]:WMODe*).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns SIFI or DIR.

Examples

```
:SOURce1:SIFI:MODE SIFI /*Enables the SiFi function for CH1.*/
:SOURce1:SIFI:MODE? /*Queries the SiFi on/off state for CH1. The
query returns SIFI.*/
```

3.11.24 [:SOURce[<n>]]:SIFI:TYPE

Syntax

[:SOURce[<*n*>]]:SIFI:TYPE <*type*>

[:SOURce[<*n*>]]:SIFI:TYPE?

Description

Sets or queries the SiFi mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<type>	Discrete	{NORMal|STEP|INTer|EDGE}	NORMal

Remarks

- **NORMal:** Normal mode. It has wide and flat frequency response as well as short edge time, but the step response produces a large overshoot.
- **STEP:** Step mode. It has more ideal step response, narrow bandwidth, longer rise/fall time, and longer edge time.
- **INTer:** Interpolation mode. It guarantees the output of signals with no distortion at all.
- **EDGE:** Edge Adjust mode. It allows you to define the edge time to create pulses with arbitrary edge time.

If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns NORM, STEP, INT, or EDGE.

Examples

```
:SOURce1:SIFI:TYPE NORMal /*Sets the SiFi mode to Normal for CH1.*/
:SOURce1:SIFI:TYPE? /*Queries the SiFi mode for CH1. The query
returns NORM.*/
```

3.11.25 [:SOURce[<n>]]:SKEW

Syntax

[:SOURce[<*n*>]]:SKEW <*skew*>

[:SOURce[<*n*>]]:SKEW?

Description

Sets or queries the skew (relative timing of the analog output) for the waveform associated with the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<skew>	Real	-2 ns to 2 ns	0 ps

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-09, indicating that the skew is 1 ns.

Examples

```
:SOURce1:SKEW 0.000000001 /*Sets the skew to 1 ns for CH1.*/
:SOURce1:SKEW? /*Queries the skew of CH1. The query returns
1.0000000000E-09.*/
```

3.11.26 [:SOURce[<n>]]:TINPut

Syntax

[:SOURce[<*n*>]]:TINPut <*trigger*>

[:SOURce[<*n*>]]:TINPut?

Description

Sets or queries the trigger source for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<trigger>	Discrete	{ITRigger|ATRigger|BTRigger}	ATRigger

Remarks

- **ITRigger** selects interrnal trigger signal as the trigger source.
- **ATRigger** selects Trig A as the trigger source.
- **BTRigger** selects Trig B as the trigger source.

If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns ITR, ATR, or BTR.

Examples

```
:SOURce1:TINPut BTRigger /*Sets the trigger source to Trig B for
CH1.*/
:SOURce1:TINPut? /*Queries the trigger source for CH1. The query
returns BTR.*/
```

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

Syntax

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

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

Description

Sets or queries the waveform amplitude for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<amp>	Real	Refer to *Remarks*	Refer to *Remarks*

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

The amplitude range and default value depend on the selected output path (*:OUTPut[<n>]:PATH*). The unit is Vpp.

- In **DC HBW**, the amplitude ranges from 350.00 mVpp to 700.0 mVpp, and the default is 350.00 mVpp.
- In **DC AMP (optional)**, the amplitude ranges from 25.00 mVpp to 1.00 Vpp, and the default is 25.00 mVpp.
- In **AC**, the amplitude ranges from 50.00 mVpp to 2.00 Vpp, and the default is 50.00 mVpp.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E+00, indicating that the amplitude is 1 Vpp.

Examples

```
:SOURce1:VOLTage:LEVel:IMMediate:AMPLitude 1 /*Sets the amplitude
to 1 Vpp for CH1.*/
:SOURce1:VOLTage:LEVel:IMMediate:AMPLitude? /*Queries the amplitude
of CH1. The query returns 1.0000000000E+00, indicating that the
amplitude of CH1 is 1 Vpp.*/
```

3.11.28 [:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:OFFSet

Syntax

[:SOURce[<*n*>]]:VOLTage[:LEVel][:IMMediate]:OFFSet <*offset*>

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

Description

Sets or queries the offset of the wave in the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<offset>	Real	Refer to *Remarks*	0 mV

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

The offset range depends on the selected output path (*:OUTPut[<n>]:PATH*).

- In **DC HBW**, the offset ranges from -20.00 mV to 20.00 mV.
- In **DC AMP (optional)**, the offset ranges from -2.0000 V to 2.0000 V.
- In **AC**, the offset ranges from -5.0000 V to 5.0000 V.

Return Format

The query returns the value in scientific notation. For example, the query might return 2.0000000000E-03, indicating that the offset is 2 mV.

Examples

```
:SOURce1:VOLTage:LEVel:IMMediate:OFFSet 0.002 /*Sets the offset to
2 mV for CH1.*/
:SOURce1:VOLTage:LEVel:IMMediate:OFFSet? /*Queries the waveform
offset for CH1. The query returns 2.0000000000E-03.*/
```

3.11.29 [:SOURce[<n>]]:WMODe

Syntax

[:SOURce[<*n*>]]:WMODe <*mode*>

[:SOURce[<*n*>]]:WMODe?

Description

Sets or queries the work mode (Real or Complex) for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<mode>	Discrete	{REAL|COMPlex}	REAL

Remarks

- The output mode can be set to either Real or Complex (optional).
  - **REAL** sets the mode to Real. The selected channel can only output real waveforms.
  - **COMPlex** sets the mode to Complex (optional). The selected channel can only output complex waveforms.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- When the clock frequency is greater than 5 GHz (only available in complex mode), the mode cannot be set to Real.

Return Format

The query returns REAL or COMP.

Examples

```
:SOURce1:WMODe COMPlex /*Sets the mode to Complex for CH1.*/
:SOURce1:WMODe? /*Queries the work mode of CH1. The query returns
COMP.*/
```

3.12 :STORage Commands

:STORage commands are used to export sequence, modulation, and waveform files as well as import sequence, modulation, waveform, and Marker files.

3.12.1 :STORage:DATA:MARKer:IMPort

Syntax

`:STORage:DATA:MARKer:IMPort` <*path*>,<*wave*>,<*type*>

Description

Imports Marker as the Marker of the specified waveform.

Parameter

Name	Type	Range	Default
<path>	ASCII String	Available path	-
<wave>	ASCII String	Available waveform name.	-
<type>	Discrete	{MK1|MK2}	MK1

Remarks

- The parameter <path> is used to set the path of the imported Marker, for example, D:/file/Marker.mkr1.
- The parameter <wave> is used to decide which waveform uses the imported Marker.
- The parameter <type> is used to set the imported Marker as Marker1 or Marker2.

Return Format

None.

Examples

```
:STORage:DATA:MARKer:IMPort D:/file/Marker.mkr1,wave,MK1 /*Sets the
imported Marker from the folder named "file" in D disk as the
Marker1 of the waveform named "wave".*/
```

3.12.2 :STORage:DATA:MODulation:EXPort

Syntax

:STORage:DATA:MODulation:EXPort <*path*>

Description

Exports the specified modulated waveform.

Parameter

Name	Type	Range	Default
<path>	ASCII string	Available path	-

Remarks

<path> specifies the path of the exported waveform. For example, the path can be D:/file/Mod.wfm.

Return Format

None.

Examples

```
:STORage:DATA:MODulation:EXPort D:/file/Mod.wfm /*Exports the
modulated waveform named "Mod" to the folder named "file" in D
disk.*/
```

3.12.3 :STORage:DATA:MODulation:IMPort

Syntax

:STORage:DATA:MODulation:IMPort <*path*>

Description

Imports the specified modulated waveform.

Parameter

Name	Type	Range	Default
<path>	ASCII string	Available path	-

Remarks

<path> specifies the path of the imported modulated waveform. For example, the path can be D:/file/Mod.

Return Format

None.

Examples

```
:STORage:DATA:MODulation:IMPort D:/file/Mod /*Imports the modulated
waveform named "Mod" from the folder named "file" in D disk.*/
```

3.12.4 :STORage:DATA:SEQuence:EXPort

Syntax

:STORage:DATA:SEQuence:EXPort <*path*>

Description

Exports the specified sequence.

Parameter

Name	Type	Range	Default
<path>	ASCII string	Available path	-

Remarks

<path> specifies the path of the exported sequence. For example, the path can be D:/file/Seq.seq.

Return Format

None.

Examples

```
:STORage:DATA:SEQuence:EXPort D:/file/Seq.seq /*Exports the
sequence named "Seq" to the folder named "file" in D disk.*/
```

3.12.5 :STORage:DATA:SEQuence:IMPort

Syntax

:STORage:DATA:SEQuence:IMPort <*path*>

Description

Imports the specified sequence.

Parameter

Name	Type	Range	Default
<path>	ASCII string	Available path	-

Remarks

- <path> specifies the path of the imported sequence. For example, the path can be D:/file/Seq.
- If the name of the sequence or its waveform or subsequence already exists, the sequence cannot be imported.

Return Format

None.

Examples

```
:STORage:DATA:SEQuence:IMPort D:/file/Seq /*Imports the sequence
named "Seq" from the folder named "file" in D disk.*/
```

3.12.6 :STORage:DATA:WAVeform:EXPort

Syntax

:STORage:DATA:WAVeform:EXPort <*path*>,<*type*>

Description

Exports the specified waveform.

Parameter

Name	Type	Range	Default
<path>	ASCII string	Available path	-
<type>	Discrete	{TXT|WFM}	-

Remarks

- <path> specifies the path of the exported waveform. For example, the path can be D:/file/wave.wfm.
- <type> specifies the file format of the exported waveform.
- The file extension is .wfm for both TXT and WFM format.

Return Format

None.

Examples

```
:STORage:DATA:WAVeform:EXPort D:/file/wave.wfm,TXT /*Exports the
waveform named "Wave" to the folder named "file" in D disk. The
file format is TXT.*/
```

3.12.7 :STORage:DATA:WAVeform:IMPort

Syntax

:STORage:DATA:WAVeform:IMPort <*path*>[,<*datatype*>[,<*eol*>[,<*type*>[,<*amp*>,[<*offset*>]]]]]

Description

Imports the specified waveform.

Parameter

Name	Type	Range	Default
<path>	ASCII string	Available path	-
<datatype>	Discrete	{NORM|VOLT}	NORM
<eol>	Discrete	{ENTer|COMMa|SEMicolon}	ENTer
<type>	Discrete	{DCODe|FLOat}	DCODe
<amp>	Real	20 mV to 2 V	500 mV
<offset>	Real	-1 V to 1 V	0 V

Remarks

<file> specifies the path of the imported waveform. For example, the path can be D:/file/wave.txt. If the imported waveform file is in TXT format, you need to configure the data type.

- <datatype> specifies the imported waveform data type.
  - **NORM** normalize the imported data.
  - **VOLT** sets the data as voltage value.
- <eol> specifies the separator between each unit.
  - **ENTer** sets the separator to enter.
  - **COMMa** sets the separator to comma ",".
  - **SEMicolon** sets the separator to semicolon ";".
- After the data type is set to "NORM", you need to set the imported waveform data format in <type>.

  - **DCODe** sets the format to hexadecimal.
  - **FLOat** sets the format to decimal.
- When the data type is set to "NORM", you also need to set the amplitude <amp> and offset <offset>. The default unit is V.

Return Format

None.

Examples

```
:STORage:DATA:WAVeform:IMPort D:/file/
Wave.txt,NORM,ENTer,DCODe,0.2,0 /*Imports the waveform named "Wave"
from the folder named "file" in D disk and sets the data type to
normalization, separator to enter, data format to hexadecimal,
amplitude to 200 mV, and offset to 0 V.*/
```

3.13 :SYSTem Commands

:SYSTem commands are used to set or query system parameters.

3.13.1 :SYSTem:BEEPer[:IMMediate]

Syntax

`:SYSTem:BEEPer[:IMMediate]`

Description

Issues a single beep immediately.

Parameter

None.

Remarks

This command is valid regardless of the beeper on/off state. This command issues a beep immediately even though the beeper is turned off.

Return Format

None.

Examples

```
:SYSTem:BEEPer:IMMediate /*Issues a single beep immediately.*/
```

3.13.2 :SYSTem:BEEPer:STATe

Syntax

`:SYSTem:BEEPer:STATe` <*state*>

`:SYSTem:BEEPer:STATe?`

Description

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

Parameter

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

Remarks

None.

Return Format

The query returns 1 or 0.

Examples

```
:SYSTem:BEEPer:STATe 1 /*Enables the beeper.*/
:SYSTem:BEEPer:STATe? /*Queries the on/off state of the beeper. The
query returns 1.*/
```

3.13.3 :SYSTem:DATE

Syntax

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

:SYSTem:DATE?

Description

Sets or queries the system date.

Parameter

Name	Type	Range	Default
<year>	Integer	1900 to 2100	-
<month>	Integer	1 to 12	-
<day>	Integer	1 to 31 (28, 29, or 30)	-

Remarks

None.

Return Format

The query returns the system date in character string. The year, month, and day are separated by "-".

Examples

```
:SYSTem:DATE 2022,05,01 /*Sets the system date to May 1, 2022.*/
:SYSTem:DATE? /*Queries the system date. The query returns
2022-05-01.*/
```

3.13.4 :SYSTem:INST:BUILd?

Syntax

`:SYSTem:INST:BUILd?`

Description

Queries the build date and time.

Parameter

None.

Remarks

None.

Return Format

The query returns the build date and time in string. For example, the query might return Apr 27 2022 19:19:26.

Examples

```
:SYSTem:INST:BUILd? /*Queries the build date and time. The query
might return Apr 27 2022 19:19:26.*/
```

3.13.5 :SYSTem:INST:HARDver?

Syntax

`:SYSTem:INST:HARDver?`

Description

Queries the instrument's hardware version.

Parameter

None.

Remarks

None.

Return Format

The query returns hardware version in string.

Examples

```
:SYSTem:INST:HARDver? /*Queries the instrument's hardware version.
The query returns 000.02.01.000.*/
```

3.13.6 :SYSTem:INST:KUC?

Syntax

:SYSTem:INST:KUC?

Description

Queries the instrument's KUC.

Parameter

None.

Remarks

None.

Return Format

The query returns the KUC in string.

Examples

```
:SYSTem:INST:KUC? /*Queries the instrument's KUC. The query might
return 20016,2022-05-20,1447.*/
```

3.13.7 :SYSTem:INST:KUM?

Syntax

:SYSTem:INST:KUM?

Description

Queries the instrument's KUM.

Parameter

None.

Remarks

None.

Return Format

The query returns the KUM in string.

Examples

```
:SYSTem:INST:KUM? /*Queries the instrument's KUM. The query might
return 20037,2022-05-20,839.*/
```

3.13.8 :SYSTem:INST:KUS?

Syntax

:SYSTem:INST:KUS?

Description

Queries the instrument's KUS.

Parameter

None.

Remarks

None.

Return Format

The query returns the KUS in string.

Examples

```
:SYSTem:INST:KUS? /*Queries the instrument's KUS. The query might
return 20037,2022-05-20,839.*/
```

3.13.9 :SYSTem:INST:MODel?

Syntax

:SYSTem:INST:MODel?

Description

Queries the instrument's model number.

Parameter

None.

Remarks

None.

Return Format

The query returns the model number in string, for example, DG70004.

Examples

```
:SYSTem:INST:MODel? /*Queries the instrument's model number. The
query returns DG70004.*/
```

3.13.10 :SYSTem:INST:SERial?

Syntax

`:SYSTem:INST:SERial?`

Description

Queries the instrument's serial number.

Parameter

None.

Remarks

None.

Return Format

The query returns the serial number in string, for example, DG7A231600001.

Examples

```
:SYSTem:INST:SERial? /*Queries the instrument's serial number. The
query returns DG7A231600001.*/
```

3.13.11 :SYSTem:INST:SOFTver?

Syntax

`:SYSTem:INST:SOFTver?`

Description

Queries the instrument's firmware version.

Parameter

None.

Remarks

None.

Return Format

The query returns firmware version in string.

Examples

```
:SYSTem:INST:SOFTver? /*Queries the instrument's firmware version.
The query might return 00.01.00.17.04.*/
```

3.13.12 :SYSTem:LANGuage

Syntax

:SYSTem:LANGuage <*language*>

:SYSTem:LANGuage?

Description

Sets or queries the system language.

Parameter

Name	Type	Range	Default
<language>	Discrete	{SCHinese|ENGLish}	-

Remarks

The language settings are not affected by factory default settings (**RST*).

Return Format

The query returns SCH or ENGL.

Examples

```
:SYSTem:LANGuage ENGLish /*Sets the system language to English.*/
:SYSTem:LANGuage? /*Queries the system language. The query returns
ENGL.*/
```

3.13.13 :SYSTem:LOCKscreen:ENABle

Syntax

:SYSTem:LOCKscreen:ENABle <*state*>

:SYSTem:LOCKscreen:ENABle?

Description

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

Parameter

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

Remarks

None.

Return Format

The query returns 1 or 0.

Examples

```
:SYSTem:LOCKscreen:ENABle ON /*Enables the screen lock function.*/
:SYSTem:LOCKscreen:ENABle? /*Queries whether the screen is locked.
The query returns 1.*/
```

3.13.14 :SYSTem:OPTion:INSTall

Syntax

:SYSTem:OPTion:INSTall <*license*>

Description

Installs the option.

Parameter

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

Remarks

- To install the option, first purchase the required option to obtain the key, and then use the key to obtain the option license according to the following steps.
  - Log in to the RIGOL official website (*http://www.rigol.com*), click **SERVICE CENTRE** > **License Activation** to enter the software license registration interface.
  - In the software license registration interface, input the correct key, serial number (click or tap ⊞ > **Utility** > **About** to obtain the serial number of the instrument), and verification code. Then click **Generate** to obtain the option license file download link. If you need to use the file, please download it to the USB storage device.
- The license is a fixed length of strings. Each instrument has a unique license.

Return Format

None.

Examples

```
None.
```

3.13.15 :SYSTem:OPTion:STATus?

Syntax

:SYSTem:OPTion:STATus? <*type*>

Compatible Command Syntax

:SYSTem:OPTion:VALid? <*type*>

Description

Queries whether an option is activated or not.

Parameter

Name	Type	Range	Default
<type>	Discrete	{DIGup|SEQ|DC|PJITTer|MTONe}	-

Remarks

- **DIGup:** Digital Up Converter (DUC) and IQ Modulation
- **SEQ:** Complex sequence function
- **DC:** DC amplifier output
- **PJITTer:** High-speed serial function
- **MTONe:** Multitone & Chirp mode

Return Format

The query returns 0 or 1. 0 indicates the option is not installed. 1 indicates that the official option has been installed.

Examples

```
:SYSTem:OPTion:STATus? SEQ /*Queries whether the Complex sequence
function is activated. The query returns 0.*/
```

Compatible Command Example

```
:SYSTem:OPTion:VALid? SEQ /*Queries whether the Complex sequence
function is activated. The query returns 0.*/
```

3.13.16 :SYSTem:OPTion:UNINstall

Syntax

:SYSTem:OPTion:UNINstall

Description

Uninstalls all the official options.

Parameter

None.

Remarks

After the options have been uninstalled, you need to restart the instrument.

Return Format

None.

Examples

```
None.
```

3.13.17 :SYSTem:POWeron

Syntax

:SYSTem:POWeron <*power_on*>

:SYSTem:POWeron?

Description

Sets or queries the system configuration type recalled by the instrument when it is powered on again after power-off.

Parameter

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

Remarks

- **DEFault** uses the factory default setting.
- **LAST** uses the last power-on configuration, including all the system parameters and states except for the channel on/off state and clock source.

Return Format

The query returns DEF or LAST.

Examples

```
:SYSTem:POWeron LAST /*Enables the instrument to recall last value
after it is powered on again.*/
:SYSTem:POWeron? /*Queries the configuration type. The query
returns LAST.*/
```

3.13.18 :SYSTem:RESet

Syntax

`:SYSTem:RESet`

Description

Restarts the instrument.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
None.
```

3.13.19 :SYSTem:SCReencap

Syntax

`:SYSTem:SCReencap`

Description

Captures the current screen.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
None.
```

3.13.20 :SYSTem:SHUTdown

Syntax

`:SYSTem:SHUTdown`

Description

Shuts down the instrument.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
None.
```

3.13.21 :SYSTem:SSAVer:PREView

Syntax

:SYSTem:SSAVer:PREView

Description

Previews the screen saver.

Parameter

None.

Remarks

None.

Return Format

None.

Examples

```
None.
```

3.13.22 :SYSTem:SSAVer:SELect

Syntax

:SYSTem:SSAVer:SELect <*type*>

:SYSTem:SSAVer:SELect?

Description

Sets or queries the state of the screen saver.

Parameter

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

Remarks

- **OFF** turns off the screen saver.
- **TEXT** sets the screen saver to "Text". After the instrument is idle and keeps a specified time (*:SYSTem:SSAVer:TIME*) of inactivity, the screen saver appears. You can use *:SYSTem:SSAVer:TEXT* to customize the text.

Return Format

The query returns OFF or TEXT.

Examples

```
:SYSTem:SSAVer:SELect TEXT /*Sets the screen saver to Text.*/
:SYSTem:SSAVer:SELect? /*Queries the screen saver. The query
returns TEXT.*/
```

3.13.23 :SYSTem:SSAVer:TEXT

Syntax

:SYSTem:SSAVer:TEXT <*string*>

:SYSTem:SSAVer:TEXT?

Description

Sets or queries the text displayed when the screen saver is enabled.

Parameter

Name	Type	Range	Default
<string>	ASCII string	-	-

Remarks

When the screen saver is enabled, the text is dynamically displayed on the screen.

Return Format

The query returns the text in string.

Examples

```
:SYSTem:SSAVer:TEXT Rigol StationMax /*Sets the text to Rigol
StationMax.*/
```

```
:SYSTem:SSAVer:TEXT? /*Queries the text. The query returns Rigol
StationMax.*/
```

3.13.24 :SYSTem:SSAVer:TIME

Syntax

:SYSTem:SSAVer:TIME <*time*>

:SYSTem:SSAVer:TIME?

Description

Sets or queries the wait time in minutes (min) of the screen saver.

Parameter

Name	Type	Range	Default
<time>	Integer	1 to 999	1

Remarks

When the screen saver function is enabled, it is activated if the instrument has been left idle for the time set in <time>.

Return Format

The query returns an integer between 1 and 999. For example, the query might return 10.

Examples

```
:SYSTem:SSAVer:TIME 10 /*Sets the wait time to 10 minutes.*/
:SYSTem:SSAVer:TIME? /*Queries the wait time. The query returns
10.*/
```

3.13.25 :SYSTem:TIME

Syntax

:SYSTem:TIME <*hour*>,<*minute*>,<*second*>

:SYSTem:TIME?

Description

Sets or queries the system time.

Parameter

Name	Type	Range	Default
<hour>	Integer	0 to 23	-

Name	Type	Range	Default
<minute>	Integer	0 to 59	-
<second>	Integer	0 to 59	-

Remarks

There is a certain delay between the return time value and the set time value due to the command response time and other factors.

Return Format

The query returns the system time in string.

Examples

```
:SYSTem:TIME 16,10,17 /*Sets the system time to 16:10:17.*/
:SYSTem:TIME? /*Queries the system time. The query returns
16:10:17.*/
```

3.13.26 :SYSTem:VERSion?

Syntax

:SYSTem:VERSion?

Description

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

Parameter

None.

Remarks

None.

Return Format

The query returns the SCPI version in string.

Examples

```
:SYSTem:VERSion? /*Queries the SCPI version. The query might return
3.0.*/
```

3.13.27 :SYSTem:VIBRation:ENABle

Syntax

:SYSTem:VIBRation:ENABle <*state*>

:SYSTem:VIBRation:ENABle?

Description

Sets or queries the on/off state of the touch vibration.

Parameter

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

Remarks

When the touch vibration is turned on, the 3.5-inch secondary screen will vibrate when it is used to operate the instrument.

Return Format

The query returns 1 or 0.

Examples

```
:SYSTem:VIBRation:ENABle 1 /*Turns on the touch vibration.*/
:SYSTem:VIBRation:ENABle? /*Queries the on/off state of the touch
vibration. The query returns 1.*/
```

3.13.28 :SYSTem:VIBRation[:IMMediate]

Syntax

`:SYSTem:VIBRation[:IMMediate]`

Description

Generates a touch vibration immediately.

Parameter

None.

Remarks

This command triggers a touch vibration immediately for the secondary screen when the touch vibration function is enabled (*:SYSTem:VIBRation:ENABle*).

Return Format

None.

Examples

```
:SYSTem:VIBRation:IMMediate /*Generates a touch vibration for the
secondary screen.*/
```

3.13.29 :SYSTem:ZBDBackctrl:STATe

Syntax

:SYSTem:ZBDBackctrl:STATe <*state*>

:SYSTem:ZBDBackctrl:STATe?

Description

Sets or queries the on/off state of ZBD backlight.

Parameter

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

Remarks

1 or ON turns on the backlight of the front-panel electronic label.

Return Format

The query returns 1 or 0.

Examples

```
:SYSTem:ZBDBackctrl:STATe 1 /*Turns on the ZBD backlight.*/
:SYSTem:ZBDBackctrl:STATe? /*Queries the on/off state of ZBD
backlight. The query returns 1.*/
```

3.14 :TRIGger Commands

:TRIGger commands are used to set or query the trigger interval, external trigger input level (threshold), trigger mode, and polarity of the external trigger slope, query the external trigger impedance, and generate a trigger event.

3.14.1 :TRIGger[:IMMediate]

Syntax

:TRIGger[:IMMediate] <*trigger*>

Description

Generates a trigger A or B event.

Parameter

Name	Type	Range	Default
<trigger>	Discrete	{ATRigger|BTRigger}	ATRigger

Remarks

None.

Return Format

None.

Examples

```
:TRIGger:IMMediate ATRigger /*Generates a trigger A event.*/
```

3.14.2 :TRIGger:IMPedance?

Syntax

:TRIGger:IMPedance? [<*trigger*>]

Description

Queries the external trigger impedance.

Parameter

Name	Type	Range	Default
<trigger>	Discrete	{ATRigger|BTRigger}	ATRigger

Remarks

None.

Return Format

The query returns 1.0000000000E+06, indicating that the external trigger impedance is 1 MΩ.

Examples

```
:TRIGger:IMPedance? ATRigger /*Queries the Trig A impedance. The
query returns 1.0000000000E+06.*/
```

3.14.3 :TRIGger:INTerval

Syntax

:TRIGger:INTerval <*interval*>

:TRIGger:INTerval?

Description

Sets or queries the internal trigger interval.

Parameter

Name	Type	Range	Default
<interval>	Real	10 μs to 10 s	10 μs

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-03, indicating that the internal trigger interval is 1 ms.

Examples

```
:TRIGger:INTerval 0.001 /*Sets the internal trigger interval to 1
ms.*/
:TRIGger:INTerval? /*Queries the internal trigger interval. The
query returns 1.0000000000E-03.*/
```

3.14.4 :TRIGger:LEVel

Syntax

:TRIGger:LEVel <*level*>[,<*trigger*>]

:TRIGger:LEVel? [<*trigger*>]

Description

Sets or queries the external trigger input level (threshold).

Parameter

Name	Type	Range	Default
<level>	Real	-5.00 V to 5.00 V	0 V
<trigger>	Discrete	{ATRigger|BTRigger}	ATRigger

Remarks

The external trigger input signal must cross the set trigger level for a trigger event to occur.

Return Format

The query returns the value in scientific notation. For example, the query may return 1.0000000000E+00, indicating that the trigger input level is 1 V.

Examples

```
:TRIGger:LEVel 1,ATRigger /*Sets the Trig A level to 1 V.*/
:TRIGger:LEVel? ATRigger /*Queries the Trig A level. The query
returns 1.0000000000E+00.*/
```

3.14.5 :TRIGger:MODE

Syntax

:TRIGger:MODE <*mode*>[,<*trigger*>]

:TRIGger:MODE? [<*trigger*>]

Description

Sets or queries the trigger timing (Sync or Async) used when an external trigger source is being used.

Parameter

Name	Type	Range	Default
<mode>	Discrete	{SYNChronous|ASYNchronous}	ASYNchronous
<trigger>	Discrete	{ATRigger|BTRigger}	ATRigger

Remarks

- **SYNChronous** triggering slows the trigger clock rate to provide a longer setup time, making it easier to align timing events between instruments. This is the recommended trigger type when using the Sync Clock Out to synchronize with external devices.
- **ASYNchronous** triggering provides the smallest delay between the trigger event and starting the waveform playout. This is the fastest triggering type.
- If [<trigger>] is omitted, it is interpreted as ATRigger.

Return Format

The query returns SYNC or ASYN.

Examples

```
:TRIGger:MODE SYNChronous,ATRigger /*Sets the trigger timing for
Trig A to synchronous.*/
:TRIGger:MODE? ATRigger /*Queries the trigger timing for Trig A.
The query returns SYNC.*/
```

3.14.6 :TRIGger:SLOPe

Syntax

:TRIGger:SLOPe <*slope*>[,<*trigger*>]

:TRIGger:SLOPe? [<*trigger*>]

Description

Sets or queries the polarity of the external trigger slope.

Parameter

Name	Type	Range	Default
<slope>	Discrete	{POSitive|NEGative}	POSitive
<trigger>	Discrete	{ATRigger|BTRigger}	ATRigger

Remarks

- **POSitive** specifies a trigger on the "rising edge" of the external trigger input level (*:TRIGger:LEVel*).
- **NEGative** specifies a trigger on the "falling edge" of the external trigger input level.
- If [<trigger>] is omitted, it is interpreted as ATRigger.

Return Format

The query returns POS or NEG.

Examples

```
:TRIGger:SLOPe POSitive,ATRigger /*Selects the rising edge for
Trigger A.*/
:TRIGger:SLOPe? ATRigger /*Queries the external Trigger A slope.
The query returns POS.*/
```

3.15 :WLISt Commands

:WLISt commands are used to query waveform parameters in waveform list or create new waveforms in Table Editor.

3.15.1 :WLISt:LAST?

Syntax

:WLISt:LAST?

Description

Queries the name of the most recently added waveform in the waveform list.

Parameter

None.

Remarks

None.

Return Format

The query returns strings.

Example

```
:WLISt:LAST? /*Queries the name of the last waveform added. For
example, the query might return wave_1.*/
```

3.15.2 :WLISt:LIST?

Syntax

:WLISt:LIST?

Description

Queries all waveform names in the waveform list.

Parameter

None.

Remarks

None.

Return Format

The query returns a string, for example, wave_1,wave_2,wave_3.

Examples

```
:WLISt:LIST? /*Queries a list of all waveform names in the waveform
list. The query might return wave_1,wave_2,wave_3.*/
```

3.15.3 :WLISt:NAME?

Syntax

:WLISt:NAME? <*index*>

Description

Queries the waveform name at the position specified by the index value in the waveform list.

Parameter

Name	Type	Range	Default
<index>	Integer	NR1	-

Remarks

You can use *:WLISt:SIZE?* to query the number of waveforms in waveform list.

Return Format

The query returns a string. For example, the query might return wave_2. If no waveform exists at the specified position, the query returns a null string.

Examples

```
:WLISt:NAME? 2 /*Queries the name of the second waveform in the
waveform list. The query might return wave_2.*/
```

3.15.4 :WLISt:SIZE?

Syntax

:WLISt:SIZE?

Description

Queries the number of waveforms in waveform list.

Parameter

None.

Remarks

None.

Return Format

The query returns an integer, for example, 7.

Examples

```
:SLISt:SIZE? /*Queries the total number of waveforms in waveform
list. The query might return 7.*/
```

3.15.5 :WLISt:WAVeform:CYCLe?

Syntax

:WLISt:WAVeform:CYCLe? <*wfm_name*>

Description

Queries the Cycles value (number of times the waveform repeats) of the named waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is available for Sine, Square, and Triangle.

Return Format

The query returns the value in scientific notation. For example, the query might return 2.4000000000E+03, indicating that the Cycles value is 2400.

Examples

```
:WLISt:WAVeform:CYCLe? Wave /*Queries the Cycles value of the
waveform named "Wave". The query might return 2.4000000000E+03.*/
```

3.15.6 :WLISt:WAVeform:DATA

Syntax

:WLISt:WAVeform:DATA <*wfm_name*>,<*startIndex*>,<*size*>,<*block_data*>

:WLISt:WAVeform:DATA? <*wfm_name*>[,<*startIndex*>[,<*size*>]]

Description

Sets or queries the waveform data.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-
<startIndex>	Integer	Refer to *Remarks*	-
<size>	Integer	Refer to *Remarks*	-
<block_data>	IEEE 488.2 block	Refer to *Remarks*	-

Remarks

- You can use this command to set or query a total number of <size> points starting from <startIndex>. startIndex+size-1 ≤ waveform length. The minimum <size> is 1.
- <block_data> is used to set the waveform data. The data is transferred in the format of TMC header+waveform data point. The TMC header is in #NXXXXX

format; wherein, # is the TMC header identifier; N following # represents the length of the character string which describes the waveform length; the length of the waveform data points is expressed in ASCII strings. For example, the data read for one time is #9000000200XXXX.... It indicates the 9 bytes are used to describe the data length. 000000200 indicates the length of waveform data, that is, 200 bytes.

- You can use *:WLISt:WAVeform:MARKer[<n>]:DATA* to set or query the Marker data.
- When the waveform is IQ wave, please use *:WLISt:WAVeform:DATA:I* and *:WLISt:WAVeform:DATA:Q* to set or query the I data and Q data respectively.

Return Format

The query returns a binary stream.

Examples

```
:WLISt:WAVeform:DATA Wave,1,100,#9000000200xxxx... /*Edits the data
of the waveform named "Wave". The data size is 100 (200 bytes) and
the start index is 1 (the first point).*/
:WLISt:WAVeform:DATA? Wave,1,100 /*Queries the data of the waveform
named "Wave". The data size is 100 and the start index is 1.*/
```

3.15.7 :WLISt:WAVeform:DATA:I

Syntax

:WLISt:WAVeform:DATA:I <*wfm_name*>,<*startIndex*>,<*size*>,<*block_data*>

:WLISt:WAVeform:DATA:I? <*wfm_name*>[,<*startIndex*>[,<*size*>]]

Description

Sets or queries I data.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-
<startIndex>	Integer	Refer to *Remarks*	-
<size>	Integer	Refer to *Remarks*	-
<block_data>	IEEE 488.2 block	Refer to *Remarks*	-

Remarks

- You can use this command to set or query a total number of <size> points starting from <startIndex>. startIndex+size-1 ≤ waveform length. The minimum <size> is 1.
- <block_data> is used to set the waveform data. The data is transferred in the format of TMC header+waveform data point. The TMC header is in #NXXXXX format; wherein, # is the TMC header identifier; N following # represents the length of the character string which describes the waveform length; the length of the waveform data points is expressed in ASCII strings. For example, the data read for one time is #9000000200XXXX.... It indicates the 9 bytes are used to describe the data length. 000000200 indicates the length of waveform data, that is, 200 bytes.
- You can use *:WLISt:WAVeform:MARKer[<n>]:DATA* to set or query the Marker data.
- You can use *:WLISt:WAVeform:DATA* to set or query the real waveform data.

Return Format

The query returns a binary stream.

Examples

```
:WLISt:WAVeform:DATA:I Wave,1,100,#9000000200xxxx... /*Edits the I
data of the waveform named "Wave". The data size is 100 (200 bytes)
and the start index is 1 (the first point).*/
:WLISt:WAVeform:DATA:I? Wave,1,100 /*Queries the I data of the
waveform named "Wave". The data size is 100 and the start index is
1.*/
```

3.15.8 :WLISt:WAVeform:DATA:Q

Syntax

:WLISt:WAVeform:DATA:Q <*wfm_name*>,<*startIndex*>,<*size*>,<*block_data*>

:WLISt:WAVeform:DATA:Q? <*wfm_name*>[,<*startIndex*>[,<*size*>]]

Description

Sets or queries Q data.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-
<startIndex>	Integer	Refer to *Remarks*	-

Name	Type	Range	Default
<size>	Integer	Refer to *Remarks*	-
<block_data>	IEEE 488.2 block	Refer to *Remarks*	-

Remarks

- You can use this command to set or query a total number of <size> points starting from <startIndex>. startIndex+size-1 ≤ waveform length. The minimum <size> is 1.
- <block_data> is used to set the waveform data. The data is transferred in the format of TMC header+waveform data point. The TMC header is in #NXXXXX format; wherein, # is the TMC header identifier; N following # represents the length of the character string which describes the waveform length; the length of the waveform data points is expressed in ASCII strings. For example, the data read for one time is #9000000200XXXX.... It indicates the 9 bytes are used to describe the data length. 000000200 indicates the length of waveform data, that is, 200 bytes.
- You can use *:WLISt:WAVeform:MARKer[<n>]:DATA* to set or query the Marker data.
- You can use *:WLISt:WAVeform:DATA* to set or query the real waveform data.

Return Format

The query returns a binary stream.

Examples

```
:WLISt:WAVeform:DATA:Q Wave,1,100,#9000000200xxxx... /*Edits the Q
data of the waveform named "Wave". The data size is 100 (200 bytes)
and the start index is 1 (the first point).*/
:WLISt:WAVeform:DATA:Q? Wave,1,100 /*Queries the Q data of the
waveform named "Wave". The data size is 100 and the start index is
1.*/
```

3.15.9 :WLISt:WAVeform:DELete

Syntax

:WLISt:WAVeform:DELete <*wfm*>

Description

Deletes a single waveform from the waveform list.

Parameter

Name	Type	Range	Default
<wfm>	ASCII string	Available waveform name	-

Remarks

If the deleted waveform is currently loaded into the channel, it is unloaded when this command is executed.

Return Format

None.

Examples

```
:WLISt:WAVeform:DELete Wave /*Deletes the waveform named "Wave"
from the waveform list.*/
```

3.15.10 :WLISt:WAVeform:DELete:ALL

Syntax

:WLISt:WAVeform:DELete:ALL

Description

Deletes all waveforms from the waveform list.

Parameter

None.

Remarks

None.

Return Format

None.

Example

```
:WLISt:WAVeform:DELete:ALL /*Deletes all waveforms.*/
```

3.15.11 :WLISt:WAVeform:DURation?

Syntax

:WLISt:WAVeform:DURation? <*wfm_name*>

Description

Queries the duration of the named waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is available for DC and Noise.

Return Format

The query returns the duration in scientific notation. For example, the query may return 2.4000000000E-06, indicating that the duration is 2.4 μs.

Examples

```
:WLISt:WAVeform:DURation? Wave /*Queries the duration of the
waveform named "Wave". The query might return 2.4000000000E-06.*/
```

3.15.12 :WLISt:WAVeform:DUTY?

Syntax

:WLISt:WAVeform:DUTY? <*wfm_name*>

Description

Queries the duty cycle of the named waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is available only for square wave.

Return Format

The query returns a real number. For example, the query might return 50.000000, indicating that the duty cycle is 50%.

Examples

```
:WLISt:WAVeform:DUTY? Wave /*Queries the duty cycle of the waveform
named "Wave". The query might return 50.000000.*/
```

3.15.13 :WLISt:WAVeform:FREQuency?

Syntax

:WLISt:WAVeform:FREQuency? <*wfm_name*>

Description

Queries the frequency of the named waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is available for Sine, Square, and Triangle.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E+09, indicating that the frequency is 1 GHz.

Examples

```
:WLISt:WAVeform:FREQuency? Wave /*Queries the frequency of the
waveform named "Wave". The query might return 1.0000000000E+09.*/
```

3.15.14 :WLISt:WAVeform:HIGH?

Syntax

:WLISt:WAVeform:HIGH? <*wfm_name*>

Description

Queries the high voltage level of the specified waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is not available for DC.

Return Format

The query returns a real number. For example, the query might return 0.250000, indicating that the high voltage level is 0.25 V.

Examples

```
:WLISt:WAVeform:HIGH? Wave /*Queries the high voltage level of the
waveform named "Wave". The query might return 0.250000.*/
```

3.15.15 :WLISt:WAVeform:LENGth?

Syntax

:WLISt:WAVeform:LENGth? <*wfm_name*>

Description

Queries the number of data points (not bytes) of the specified waveform in the waveform list.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

None.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E+03, indicating that the waveform contains 5000 data points.

Examples

```
:WLISt:WAVeform:LENGth? Wave /*Queries the number of data points of
the waveform named "Wave". The query might return 5.0000000000E
+03.*/
```

3.15.16 :WLISt:WAVeform:LMAXimum?

Syntax

:WLISt:WAVeform:LMAXimum?

Description

Queries the maximum number of waveform sample points allowed.

Parameter

None.

Remarks

None.

Return Format

The query returns an integer 1610612736.

Examples

```
:WLISt:WAVeform:LMAXimum? /*Queries the maximum number of waveform
sample points allowed. The query returns 1610612736.*/
```

3.15.17 :WLISt:WAVeform:LMINimum?

Syntax

`:WLISt:WAVeform:LMINimum?`

Description

Queries the minimum number of waveform sample points required for a valid waveform.

Parameter

None.

Remarks

None.

Return Format

The query returns an integer 2400.

Examples

```
:WLISt:WAVeform:LMINimum? /*Queries the minimum number of waveform
sample points required. The query returns 2400.*/
```

3.15.18 :WLISt:WAVeform:LOW?

Syntax

`:WLISt:WAVeform:LOW?` <*wfm_name*>

Description

Queries the low voltage level of the specified waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is not available for DC.

Return Format

The query returns a real number. For example, the query might return -0.250000, indicating that the low voltage level is -0.25 V.

Examples

```
:WLISt:WAVeform:LOW? Wave /*Queries the low voltage level of the
waveform named "Wave". The query might return -0.250000.*/
```

3.15.19 :WLISt:WAVeform:LOOPwidth?

Syntax

:WLISt:WAVeform:LOOPwidth? <*wfm_name*>

Description

Queries the loop width of the named waveform (Sinc).

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

None.

Return Format

The query returns a real number. For example, the query might return 50.000000, indicating that the loop width is 50%.

Examples

```
:WLISt:WAVeform:LOOPwidth? Wave /*Queries the loop width of the
waveform named "Wave". The query might return 50.000000.*/
```

3.15.20 :WLISt:WAVeform:MARKer[<n>]:DATA

Syntax

:WLISt:WAVeform:MARKer[<*n*>]:DATA <*wfm_name*>,<*startIndex*>,<*size*>,<*block_data*>

:WLISt:WAVeform:MARKer[<*n*>]:DATA? <*wfm_name*>[,<*startIndex*>[,<*size*>]]

Description

Sets or queries the Marker data.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<wfm_name>	ASCII string	Available waveform name	-
<startIndex>	Integer	Refer to *Remarks*	-
<size>	Integer	Refer to *Remarks*	-
<block_data>	IEEE 488.2 block	Refer to *Remarks*	-

Remarks

- You can use this command to set or query a total number of <size> points starting from <startIndex>. startIndex+size-1 ≤ waveform length. The minimum <size> is 1.
- <block_data> is used to set the Marker data. The data is transferred in the format of TMC header+waveform data point. The TMC header is in #NXXXXX format; wherein, # is the TMC header identifier; N following # represents the length of the character string which describes the marker length; the length of the marker data points is expressed in ASCII strings. For example, the data read for one time is #900000000208000.... It indicates the 9 bytes are used to describe the data length. 000000020 indicates the length of marker data, that is, 20 bytes. 8000...indicates that the first Marker is high level (80) and the second Marker is low level (00).
- You can use *:WLISt:WAVeform:DATA* to set or query the real waveform data.
- When the waveform is IQ wave, you can use *:WLISt:WAVeform:DATA:I* and *:WLISt:WAVeform:DATA:Q* to set or query the I data and Q data respectively.

Return Format

The query returns a binary stream.

Examples

```
:WLISt:WAVeform:MARKer1:DATA Wave,1,20,#9000000020xxxx... /*Edits
the marker1 data of the waveform named "Wave". The data size is 20
(20 bytes) and the start index is 1 (the first point).*/
:WLISt:WAVeform:MARKer1:DATA? Wave,1,20 /*Queries the marker1 data
of the waveform named "Wave". The data size is 20 and the start
index is 1.*/
```

3.15.21 :WLISt:WAVeform:MIQ

Syntax

:WLISt:WAVeform:MIQ <*I_wfm_name*>,<*Q_wfm_name*>[,<*iq_name*>]

Description

Creates an IQ waveform from two real waveforms.

Parameter

Name	Type	Range	Default
<I_wfm_name>	ASCII string	Available waveform name	-
<Q_wfm_name>	ASCII string	Available waveform name	-
<iq_name>	ASCII string	Available waveform name	-

Remarks

- The selected I wave and Q wave should be of equal length.
- <I_wfm_name> is the name of the I component. <Q_wfm_name> is the name of the Q component. <iq_name> specifies the name of the created IQ waveform. If <iq_name> is omitted, the name of the IQ wave is created based on that of the I component.

Return Format

None.

Examples

```
:WLISt:WAVeform:MIQ Wave1,Wave2,WaveIQ /*Creates an IQ waveform
named "WaveIQ" using "Wave1" as the I component and "Wave2" as the
Q component.*/
```

3.15.22 :WLISt:WAVeform:NEW

Syntax

:WLISt:WAVeform:NEW <*wfm_name*>,<*size*>[,<*format*>]

Description

Create a new waveform in Table Editor.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-
<size>	Integer	2400 to 1610612736	-
<format>	Discrete	{REAL|IQ}	REAL

Remarks

<wfm_name> specifies the name of the new waveform. <size> specifies the number of sample points. <format> specifies the type of the created waveform.

- **REAL** creates real waveforms.
- **IQ** creates IQ waveforms.

By default, the sample data of the waveform created by this command is 0. You can use *:WLISt:WAVeform:DATA* to set the real data,
and *:WLISt:WAVeform:DATA:I*/*:WLISt:WAVeform:DATA:Q* to set the I/Q data.

Return Format

None.

Examples

```
:WLISt:WAVeform:NEW Wave,2400,REAL /*Creates a new real waveform
named "Wave" with 2400 sample points.*/
```

3.15.23 :WLISt:WAVeform:PEAKpos?

Syntax

:WLISt:WAVeform:PEAKpos? <*wfm_name*>

Description

Queries the Peak Position for the specified waveform (Sinc).

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

None.

Return Format

The query returns a real number. For example, the query might return 50.000000, indicating that the peak position is 50%.

Examples

```
:WLISt:WAVeform:PEAKpos? Wave /*Queries the peak position for the
waveform named "Wave". The query might return 50.000000.*/
```

3.15.24 :WLISt:WAVeform:PHASe?

Syntax

`:WLISt:WAVeform:PHASe?` <*wfm_name*>

Description

Queries the phase of the named waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is available for Sine, Square, and Triangle.

Return Format

The query returns a real number. For example, the query might return 90.000000, indicating that the phase is 90°.

Examples

```
:WLISt:WAVeform:PHASe? Wave /*Queries the phase of the waveform
named "Wave". The query might return 90.000000.*/
```

3.15.25 :WLISt:WAVeform:SRATe?

Syntax

`:WLISt:WAVeform:SRATe?` <*wfm_name*>

Description

Queries the sample rate of the specified waveform in the waveform list.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

None.

Return Format

The query returns the sample rate in scientific notation. For example, the query might return 5.0000000000E+09, indicating the sample rate is 5 GSa/s.

Examples

```
:WLISt:WAVeform:SRATe? Wave /*Queries the sample rate of the
waveform named "Wave". The query might return 5.0000000000E+09.*/
```

3.15.26 :WLISt:WAVeform:SYMM?

Syntax

:WLISt:WAVeform:SYMM? <*wfm_name*>

Description

Queries the symmetry of the named waveform (Triangle).

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

This command is available only for Triangle waveforms.

Return Format

The query returns a real number. For example, the query might return 50.000000, indicating that the symmetry is 50%.

Examples

```
:WLISt:WAVeform:SYMM? Wave /*Queries the symmetry of the waveform
named "Wave". The query might return 50.000000.*/
```

3.15.27 :WLISt:WAVeform:TSTamp?

Syntax

:WLISt:WAVeform:TSTamp? <*wfm_name*>

Description

Queries the timestamp of the named waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

The command queries the timestamp that indicates when the waveform was created or last modified. The timestamp is updated whenever the waveform is created (added) or changed.

Return Format

The query returns the timestamp as a string in the form of "yyyy-ll-dd hh:mm:ss". Where:

• yyyy refers to a four-digit year number.

• ll refers to two-digit month number.

• dd refers to two-digit day number in the month.

• hh refers to two-digit hour number.

• mm refers to two-digit minute number.

• ss refers to two-digit second number.

Examples

```
:WLISt:WAVeform:TSTamp? Wave /*Queries the timestamp of the
waveform named "Wave". The query might return 2022-01-18 11:05:21.*/
```

3.15.28 :WLISt:WAVeform:TYPE?

Syntax

:WLISt:WAVeform:TYPE? <*wfm_name*>

Description

Queries the type of the named waveform.

Parameter

Name	Type	Range	Default
<wfm_name>	ASCII string	Available waveform name	-

Remarks

None.

Return Format

The query returns REAL (real waveform) or IQ (IQ waveform).

Examples

```
:WLISt:WAVeform:TYPE? Wave /*Queries the type of the waveform named
"Wave". The query returns REAL.*/
```

4 AFG commands

This chapter introduces the syntax, functions, parameters, and usage of each DG70000 (AFG) command. For commands related to system settings, please refer to *AWG Commands*.

CAUTION

1. **Unless otherwise specified, the descriptions in this manual take DG70004 as an example.**
2. **For the parameter setting command (time, frequency, amplitude, etc.), the instrument can only recognize the numbers, unable to recognize the unit sent together with them. For the default units of various parameters, refer to the descriptions for the specified command.**

4.1 :CLOCk Commands

:CLOCk commands are used to set and query information related to the clock.

4.1.1 :CLOCk:ROSCillator:SOURce

Syntax

```
:CLOCk:ROSCillator:SOURce <source>
:CLOCk:ROSCillator:SOURce?
```

Description

Sets or queries the clock source in AFG mode.

Parameter

Name	Type	Range	Default
<source>	Discrete	{INTernal|EXTernal}	INTernal

Remarks

- **INTernal** sets the clock source to internal reference clock.
- **EXTernal** sets the clock source to external reference clock. The instrument accepts external clock source input from the rear-panel **[EXT REF IN]** connector.

Return Format

The query returns INT or EXT.

Examples

```
:CLOCk:ROSCillator:SOURce EXTernal /*Sets the clock source in AFG
mode to external reference clock.*/
:CLOCk:ROSCillator:SOURce? /*Queries the clock source in AFG mode.
The query returns EXT.*/
```

4.2 :OUTPut Commands

:OUTPut commands are used to set and query the channel outputs.

4.2.1 :OUTPut[:STATe]

Syntax

:OUTPut[:STATe] <*state*>

:OUTPut[:STATe]?

Description

Sets or queries the output state of all channels.

Parameter

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

Remarks

1 or ON turns on all channel outputs. 0 or OFF turns off all channel outputs.

Return Format

The query returns 0 or 1.

Examples

```
:OUTPut:STATe 1 /*Turns on all channels.*/
:OUTPut:STATe? /*Queries the output state of all channels. The
query returns 1.*/
```

4.2.2 :OUTPut[<n>]:CHAN:INVerted

Syntax

:OUTPut[<*n*>]:CHAN:INVerted <*polarity*>

:OUTPut[<*n*>]:CHAN:INVerted?

Description

Sets or queries the waveform invert on/off state in the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<polarity>	Bool	{0|1|ON|OFF}	0

Remarks

- 0 or OFF sets the output mode to normal while 1 or ON sets the output mode to inverted. The waveform is inverted relative to the offset voltage.
- [<n>] determines the channel number. If omitted, interpreted as 1.

Return Format

The query returns 0 or 1.

Examples

```
:OUTPut1:CHAN:INVerted 1 /*Enables the waveform invert in CH1.*/
:OUTPut1:CHAN:INVerted? /*Queries the waveform invert on/off state
in CH1.*/
```

4.2.3 :OUTPut[<n>]:CHAN[:STATe]

Syntax

:OUTPut[<*n*>]:CHAN[:STATe] <*state*>

:OUTPut[<*n*>]:CHAN[:STATe]?

Description

Sets or queries the output state for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<state>	Bool	{0|1|OFF|ON}	0

Remarks

- 1 or ON enables the outputs for the specified channel while 0 or OFF disables the outputs for the specified channel.
- When [<n>] is omitted, it is interpreted as 1.

Return Format

The query returns 0 or 1.

Examples

```
:OUTPut1:CHAN:STATe 1 /*Enables the CH1 outputs.*/
:OUTPut1:CHAN:STATe? /*Queries the output state of CH1. The query
returns 1.*/
```

4.2.4 :OUTPut[<n>]:CHAN:VOLLimit:HIGH

Syntax

:OUTPut[<*n*>]:CHAN:VOLLimit:HIGH <*high*>

:OUTPut[<*n*>]:CHAN:VOLLimit:HIGH?

Description

Sets or queries the high limit value for output voltage for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<high>	Real	-987.5 mV to 1.5 V	250 mV

Remarks

- Voltage limit high value ≥ high level setting value.
- [<n>] determines the channel number. If omitted, interpreted as 1.

Return Format

The query returns the value in scientific notation. For example, the query may return 1.5000000000E-01, indicating that the high limit value is 150 mV.

Examples

```
:OUTPut1:CHAN:VOLLimit:HIGH 0.15 /*Sets the high limit to 150 mV
for CH1.*/
:OUTPut1:CHAN:VOLLimit:HIGH? /*Queries the high limit of CH1. The
query returns 1.5000000000E-01.*/
```

4.2.5 :OUTPut[<n>]:CHAN:VOLLimit:LOW

Syntax

:OUTPut[<*n*>]:CHAN:VOLLimit:LOW <*low*>

:OUTPut[<*n*>]:CHAN:VOLLimit:LOW?

Description

Sets or queries the low limit value for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<low>	Real	-1.5 V to 987.5 mV	-250 mV

Remarks

- Voltage limit low value ≤ low level setting value.
- [<n>] determines the channel number. If omitted, interpreted as 1.

Return Format

The query returns the value in scientific notation. For example, the query may return -1.5000000000E-01, indicating that the low limit value is -150 mV.

Examples

```
:OUTPut1:CHAN:VOLLimit:LOW -0.15 /*Sets the low level limit of CH1
to -150 mV.*/
:OUTPut1:CHAN:VOLLimit:LOW? /*Queries the low level limit of CH1.
The query returns -1.5000000000E-01.*/
```

4.2.6 :OUTPut[<n>]:CHAN:VOLLimit[:STATe]

Syntax

:OUTPut[<*n*>]:CHAN:VOLLimit[:STATe] <*state*>

:OUTPut[<*n*>]:CHAN:VOLLimit[:STATe]?

Description

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

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<state>	Bool	{0|1|ON|OFF}	0

Remarks

- Turning on the Voltage Limit function will limit the high and low levels for the specified channel. You can send *:OUTPut[<n>]:CHAN:VOLLimit:HIGH* and *:OUTPut[<n>]:CHAN:VOLLimit:LOW* to set your desired limit values for high and low level.

- [<n>] determines the channel number. If omitted interpreted as 1.

Return Format

The query returns 0 or 1.

Examples

```
:OUTPut1:CHAN:VOLLimit:STATe 1 /*Enables the Voltage Limit for 
CH1.*/
:OUTPut1:CHAN:VOLLimit:STATe? /*Queries the Voltage Limit on/off 
state of CH1. The query returns 1.*/
```

4.3 :SOURce Commands

:SOURce commands are used to set or query channel parameters.

The table below shows the frequency and period range for different waveforms in AFG mode.

Table 4.8 The frequency and period range for basic wave

Function	Frequency Range	Period Range
Sine	1 μHz to 2 GHz	500 ps to 1000 ks
Square	1 μHz to 500 MHz	2 ns to 1000 ks
Ramp	1 μHz to 50 MHz	20 ns to 1000 ks
Pulse	1 μHz to 500 MHz	2 ns to 1000 ks
Noise	N/A	N/A

4.3.1 [:SOURce[<n>]]:BURSt

4.3.1.1 [:SOURce[<n>]]:BURSt:MODE

Syntax

[:SOURce[<*n*>]]:BURSt:MODE <*type*>

[:SOURce[<*n*>]]:BURSt:MODE?

Description

Sets or queries the burst type for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<type>	Discrete	{TRIGgered|INFinity}	TRIGgered

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

When the output mode (*[:SOURce[<n>]]:MODE:FUNCtion*) is set to Burst (BURSt), you can use this command to set the burst type. DG70000 provides two options including N-Cycle Burst and Infinite Burst.

- **TRIGgered:** N-Cycle Burst (also called "triggered burst"). The instrument outputs a waveform for specified number of cycles (burst count) each time trigger is received. You can use sine, square, ramp, and pulse to create a N-Cycle burst waveform.
- **INFinity:** Infinite Burst. It sets the cycles to "Infinite ", which results in a continuous waveform once the instrument is triggered. You can use sine, square, ramp, and pulse to create an Infinite burst waveform.

Return Format

The query returns TRIG or INF.

Examples

```
:SOURce1:BURSt:MODE INFinity /*Sets the burst type to Infinite for
CH1.*/
:SOURce1:BURSt:MODE? /*Queries the burst type for CH1. The query
returns INF.*/
```

4.3.1.2 [:SOURce[<n>]]:BURSt:NCYCles

Syntax

[:SOURce[<*n*>]]:BURSt:NCYCles <*cycles*>

[:SOURce[<*n*>]]:BURSt:NCYCles?

Description

Sets or queries the number of cycles to be output in N-Cycle Burst mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<cycles>	Integer	1 to 500,000 (internal trigger) 1 to 1,000,000 (manual trigger)	1

Remarks

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- In N-Cycle Burst mode (*[:SOURce[<n>]]:BURSt:MODE*), the instrument outputs a waveform for specified number of cycles (burst count) each time trigger is received.

Return Format

The query returns an integer, for example, 500.

Examples

```
:SOURce1:BURSt:NCYCles 500 /*Sets the number of cycles to 500 for
CH1.*/
:SOURce1:BURSt:NCYCles? /*Queries the number of cycles for CH1.
The query returns 500.*/
```

4.3.1.3 [:SOURce[<n>]]:BURSt:TDELay

Syntax

[:SOURce[<*n*>]]:BURSt:TDELay <*delay*>

[:SOURce[<*n*>]]:BURSt:TDELay?

Description

Sets or queries the Delay in N-Cycle Burst or Infinite Burst mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<delay>	Real	0 ps to 500 s	0 s

Remarks

- The burst delay is only available for N-Cycle Burst and Infinite Burst mode with manual trigger. It is defined as the duration from the time when the instrument receives the trigger signal to the time when it starts to output the N-Cycle or Infinite burst.

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-01, indicating that the burst delay is 0.1 s.

Examples

```
:SOURce1:BURSt:TDELay 0.1 /*Sets the Burst Delay to 0.1 s for
CH1.*/
:SOURce1:BURSt:TDELay? /*Queries the Burst Delay for CH1. The
query returns 1.0000000000E-01.*/
```

4.3.1.4 [:SOURce[<n>]]:BURSt:IDLE

Syntax

[:SOURce[<*n*>]]:BURSt:IDLE <*idle*>

[:SOURce[<*n*>]]:BURSt:IDLE?

Description

Sets or queries the Idle Level in Burst mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<idle>	Discrete	{FPT|TOP|CENTer|BOTTom|USER}	FPT

Remarks

You have the following options:

- **FPT** selects the level at the first point of the waveform as the idle level.
- **TOP** selects the level at the top point of the waveform as the idle level.
- **CENTer** selects the level at the center point of the waveform as the idle level.
- **BOTTom** selects the level at the bottom point of the waveform as the idle level.
- **USER** selects the level at the specified point of the waveform as the idle level. You can use *[:SOURce[<n>]]:BURSt:IDLE:USER* to self-define the idle level.

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns FPT, TOP, CENT, BOTT, or USER.

Examples

```
:SOURce1:BURSt:IDLE TOP /*Sets the top point of the waveform as
the Idle Level for CH1 in Burst mode.*/
:SOURce1:BURSt:IDLE? /*Queries the Idle Level for CH1 in Burst
mode. The query returns TOP.*/
```

4.3.1.5 [:SOURce[<n>]]:BURSt:IDLE:USER

Syntax

[:SOURce[<*n*>]]:BURSt:IDLE:USER <*user*>

[:SOURce[<*n*>]]:BURSt:IDLE:USER?

Description

Sets or queries the user-defined Idle Level in Burst mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<user>	Integer	0 to 65,535	-

Remarks

- When the Idle Level (*[:SOURce[<n>]]:BURSt:IDLE*) is set to User (USER), you can use this command to self-define the idle level.
- If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns an integer between 0 and 65,535.

Examples

```
:SOURce1:BURSt:IDLE:USER 5 /*Sets the fifth point as the Idle
Level for CH1 in Burst mode.*/
:SOURce1:BURSt:IDLE:USER? /*Queries the user-defined Idle Level in
Burst mode for CH1. The query returns 5.*/
```

4.3.1.6 [:SOURce[<n>]]:BURSt:INTernal:PERiod

Syntax

[:SOURce[<*n*>]]:BURSt:INTernal:PERiod <*period*>

[:SOURce[<*n*>]]:BURSt:INTernal:PERiod?

Description

Sets or queries Burst Period of internally-triggered N-Cycle bursts for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<period>	Real	1 μs to 500 s	10 ms

Remarks

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- The burst period is defined as the time from the start of a burst to the time when the next burst starts.
- The burst period must satisfy the following formula: Burst Period ≥ Burst Count/Waveform Frequency+1 μs.
- If the burst period is too short, the instrument will automatically increase it to output with the specified burst count.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-01, indicating that the burst period is 0.1 s.

Examples

```
:SOURce1:BURSt:INTernal:PERiod 0.1 /*Sets the Burst Period of
internally-triggered N-Cycle bursts to 0.1 s for CH1.*/
:SOURce1:BURSt:INTernal:PERiod? /*Queries the Burst Period of
internally-triggered N-Cycle bursts for CH1. The query returns
1.0000000000E-01.*/
```

4.3.1.7 [:SOURce[<n>]]:BURSt:TRIGger:SOURce

Syntax

[:SOURce[<*n*>]]:BURSt:TRIGger:SOURce <*source*>

[:SOURce[<*n*>]]:BURSt:TRIGger:SOURce?

Description

Sets or queries the trigger source in N-Cycle Burst mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<source>	Discrete	{INTernal|MANual}	INTernal

Remarks

Both "Internal" and "Manual" trigger are available for the N-Cycle burst.

- **INTernal:** Internal trigger, only available for the N-Cycle burst. When internal trigger is selected, the instrument only outputs a burst of the specified cycles and the burst frequency is determined by the Burst Period (*[:SOURce[<n>]]:BURSt:INTernal:PERiod*). After completing the specified number of cycles, the instrument stops and waits for the next trigger event.
- **MANual:** Manual trigger. The instrument initiates a burst immediately each time you click or tap the Manual Trig key or send the manual trigger command (*[:SOURce[<n>]]:BURSt:TRIGger[:IMMediate]*). The trigger is ignored if the selected channel output is not enabled.

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns INT or MAN.

Examples

```
:SOURce1:BURSt:TRIGger:SOURce MANual /*Sets the trigger source to
manual trigger in N-Cycle Burst mode for CH1.*/
:SOURce1:BURSt:TRIGger:SOURce? /*Queries the trigger source in N-
Cycle Burst mode for CH1. The query returns MAN.*/
```

4.3.1.8 [:SOURce[<n>]]:BURSt:TRIGger[:IMMediate]

Syntax

[:SOURce[<*n*>]]:BURSt:TRIGger[:IMMediate]

Description

Generates a trigger event immediately. This command functions the same as clicking or tapping the manual trigger button.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Remarks

If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

None.

Examples

```
None.
```

4.3.2 [:SOURce[<n>]]:FREQuency

4.3.2.1 [:SOURce[<n>]]:FREQuency:CENTer

Syntax

[:SOURce[<*n*>]]:FREQuency:CENTer <*frequency*>

[:SOURce[<*n*>]]:FREQuency:CENTer?

Description

Sets or queries the Center Frequency in Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	Refer to *Remarks*	550 Hz

Remarks

- The instrument can sweep sine, square, and ramp. The available center frequency depends on the selected carrier wave type.
  - Sine: 1 μHz to 1 GHz
  - Square: 1 μHz to 500 MHz
  - Ramp: 1 μHz to 50 MHz
- You can set the sweep frequency boundaries using the center frequency and frequency span (*[:SOURce[<n>]]:FREQuency:SPAN*). The center frequency and frequency span differs for different waves. Also, they are limited to each other.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Center Frequency=(Start Frequency+Stop Frequency)/2. Frequency Span=Stop Frequency–Start Frequency.

- Modifying the "Center Frequency" enables the instrument to sweep from the specified "Start Frequency". The output waveform may have its amplitude characteristics changed when sweeping over a wide frequency range.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E+02, indicating that the center frequency is 500 Hz.

Examples

```
:SOURce1:FREQuency:CENTer 500 /*Sets the Center Frequency to 500
Hz for CH1.*/
:SOURce1:FREQuency:CENTer? /*Queries the Center Frequency for CH1.
The query returns 5.0000000000E+02.*/
```

4.3.2.2 [:SOURce[<n>]]:FREQuency:SPAN

Syntax

[:SOURce[<*n*>]]:FREQuency:SPAN <*frequency*>

[:SOURce[<*n*>]]:FREQuency:SPAN?

Description

Sets or queries the Frequency Span in Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	Refer to *Remarks*	900 Hz

Remarks

- You can set the sweep frequency boundaries using the center frequency (*[:SOURce[<n>]]:FREQuency:CENTer*) and frequency span. The center frequency and frequency span differs for different waves. Also, they are limited to each other. Define the maximum start/stop frequency as F_{max} and the minimum start/stop frequency as F_{min}. F_m=(F_{max}-F_{min})/2. The frequency span is affected by the center frequency: when Center Frequency < F_m, the Frequency Span is ±2×(Center Frequency-F_{min}); when Center Frequency > F_m, the Frequency Span is ±2×(F_{max}-Center Frequency).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- The instrument can sweep sine, square, and ramp.

- Center Frequency=(Start Frequency+Stop Frequency)/2. Frequency Span=Stop Frequency-Start Frequency.
- Modifying the "Frequency Span" enables the instrument to sweep from the specified "Start Frequency". The output waveform may have its amplitude characteristics changed when sweeping over a wide frequency range.

Return Format

The query returns the value in scientific notation. For example, the query might return 8.0000000000E+02, indicating that the frequency span is 800 Hz.

Examples

```
:SOURce1:FREQuency:SPAN 800 /*Sets the Frequency Span to 800 Hz
for CH1.*/
:SOURce1:FREQuency:SPAN? /*Queries the Frequency Span for CH1. The
query returns 8.000000000E+02.*/
```

4.3.2.3 [:SOURce[<n>]]:FREQuency:STARt

Syntax

[:SOURce[<*n*>]]:FREQuency:STARt <*frequency*>

[:SOURce[<*n*>]]:FREQuency:STARt?

Description

Sets or queries the Start Frequency in Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	Refer to *Remarks*	100 Hz

Remarks

- The instrument can sweep sine, square, and ramp. The available start frequency depends on the selected carrier wave type.
  - Sine: 1 μHz to 1 GHz
  - Square: 1 μHz to 500 MHz
  - Ramp: 1 μHz to 50 MHz
- The start frequency and stop frequency (*[:SOURce[n]]:FREQuency:STOP*) set the sweep's upper and lower frequency bounds. The sweep begins at the start frequency, sweeps to the stop frequency, and then resets back to the start frequency. To sweep up in frequency, set the start frequency less than the stop

frequency; to sweep down in frequency, set the opposite relationship. To sweep in a fixed frequency, set the same start frequency and stop frequency.

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Center Frequency=(Start Frequency+Stop Frequency)/2. Frequency Span=Stop Frequency–Start Frequency.
- Modifying the "Start Frequency" enables the instrument to sweep from the specified "Start Frequency". The output waveform may have its amplitude characteristics changed when sweeping over a wide frequency range.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E+02, indicating that the start frequency is 100 Hz.

Examples

```
:SOURce1:FREQuency:STARt 100 /*Sets the Start Frequency to 100 Hz
for CH1.*/
:SOURce1:FREQuency:STARt? /*Queries the Start Frequency for CH1.
The query returns 1.0000000000E+02.*/
```

4.3.2.4 [:SOURce[<n>]]:FREQuency:STOP

Syntax

[:SOURce[<*n*>]]:FREQuency:STOP <*frequency*>

[:SOURce[<*n*>]]:FREQuency:STOP?

Description

Sets or queries the Stop Frequency in Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	Refer to *Remarks*	1 kHz

Remarks

- The instrument can sweep sine, square, and ramp. The available stop frequency depends on the selected carrier wave type.
  - Sine: 1 μHz to 1 GHz
  - Square: 1 μHz to 500 MHz
  - Ramp: 1 μHz to 50 MHz
- The start frequency (*[:SOURce[<n>]]:FREQuency:STARt*) and stop frequency set the sweep's upper and lower frequency bounds. The sweep begins at the start

frequency, sweeps to the stop frequency, and then resets back to the start frequency. To sweep up in frequency, set the start frequency less than the stop frequency; to sweep down in frequency, set the opposite relationship. To sweep in a fixed frequency, set the same start frequency and stop frequency.

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Center Frequency=(Start Frequency+Stop Frequency)/2. Frequency Span=Stop Frequency–Start Frequency.
- Modifying the "Stop Frequency" enables the instrument to sweep from the specified "Start Frequency". The output waveform may have its amplitude characteristics changed when sweeping over a wide frequency range.

Return Format

The query returns the value in scientific notation. For example, the query might return 9.0000000000E+02, indicating that the stop frequency is 900 Hz.

Examples

```
:SOURce1:FREQuency:STOP 900 /*Sets the Stop Frequency to 900 Hz
for CH1.*/
:SOURce1:FREQuency:STOP? /*Queries the Stop Frequency for CH1. The
query returns 9.0000000000E+02.*/
```

4.3.2.5 [:SOURce[<n>]]:FREQuency[:FIXed]

Syntax

[:SOURce[<*n*>]]:FREQuency[:FIXed] <*frequency*>

[:SOURce[<*n*>]]:FREQuency[:FIXed]?

Description

Sets or queries the Frequency of basic wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	Refer to *Remarks*	1 kHz

Remarks

- For the frequency available for different waves, please refer to *Table 4.8 The frequency and period range for basic wave* .
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

- When the wave type is changed, the instrument still uses the frequency if the frequency is valid. Otherwise, the instrument automatically sets the frequency as the upper limit for the new wave type.

Return Format

The query returns the value in scientific notation. For example, the query might return 8.0000000000E+03, indicating that the frequency is 8 kHz.

Examples

```
:SOURce1:FREQuency:FIXed 8000 /*Sets the Frequency of basic wave
to 8 kHz for CH1.*/
:SOURce1:FREQuency:FIXed? /*Queries the Frequency of basic wave
for CH1. The query returns 8.000000000E+03.*/
```

4.3.3 [:SOURce[<n>]]:FUNCtion:SQUare:DCYCle

Syntax

[:SOURce[<*n*>]]:FUNCtion:SQUare:DCYCle <*percent*>

[:SOURce[<*n*>]]:FUNCtion:SQUare:DCYCle?

Description

Sets or queries the Duty Cycle percentage of Square wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<percent>	Real	0.001% to 99.999%	50%

Remarks

- Duty cycle represents the amount of time per period (*[:SOURce[<n>]]:PERiod[:FIXed]*) that the square wave is at a high level. The Duty Cycle is affected by Period: (500 ps/Period)*100 ≤ Duty Cycle ≤ (1-500 ps/ Period)*100
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 4.5000000000E+01, indicating that the duty cycle is 45%.

Examples

```
:SOURce1:FUNCtion:SQUare:DCYCle 45 /*Sets the Duty Cycle of Square
wave to 45% for CH1.*/
```

```
:SOURce1:FUNCtion:SQUare:DCYCle? /*Queries the Duty Cycle of Square
wave for CH1. The query returns 4.5000000000E+01.*/
```

4.3.4 [:SOURce[<n>]]:FUNCtion:RAMP:SYMMetry

Syntax

[:SOURce[<*n*>]]:FUNCtion:RAMP:SYMMetry <*symm*>

[:SOURce[<*n*>]]:FUNCtion:RAMP:SYMMetry?

Description

Sets or queries Symmetry of the Ramp for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<symm>	Real	0% to 100%	50%

Remarks

- Symmetry is defined as the percentage of the amount of time Ramp wave is rising in the period (*[:SOURce[<n>]]:PERiod[:FIXed]*).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.5000000000E+01, indicating that the symmetry is 55%.

Examples

```
:SOURce1:FUNCtion:RAMP:SYMMetry 55 /*Sets the Symmetry of Ramp to
55% for CH1.*/
:SOURce1:FUNCtion:RAMP:SYMMetry? /*Queries the Symmetry of Ramp for
CH1. The query returns 5.5000000000E+01.*/
```

4.3.5 [:SOURce[<n>]]:MODE:FUNCtion

Syntax

[:SOURce[<*n*>]]:MODE:FUNCtion <*func*>

[:SOURce[<*n*>]]:MODE:FUNCtion?

Description

Sets or queries the output mode in AFG mode.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<func>	Discrete	{BASic|BURSt|MODulation|SWEep}	BASic

Remarks

- **BASic** sets the output mode to Continuous mode.
- **BURSt** sets the output mode to Burst mode.
- **MODulation** sets the output mode to Modulation mode.
- **SWEep** sets the output mode to Sweep mode.

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns BAS, BURS, MOD, or SWE.

Examples

```
:SOURce1:MODE:FUNCtion BURSt /*Sets the output mode to Burst mode
for CH1 in AFG mode.*/
:SOURce1:MODE:FUNCtion? /*Queries the output mode for CH1 in AFG
mode. The query returns BURS.*/
```

4.3.6 [:SOURce[<n>]][:MOD]

4.3.6.1 [:SOURce[<n>]][:MOD]:AM:DEPTh

Syntax

[:SOURce[<*n*>]][:MOD]:AM:DEPTh <*depth*>

[:SOURce[<*n*>]][:MOD]:AM:DEPTh?

Description

Sets or queries the AM Modulation Depth for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<depth>	Real	0% to 120%	100%

Remarks

- Modulation depth is a percentage that represents the amplitude variation. At 0% depth, the amplitude is one-half of the carrier's amplitude setting. At 100% depth, the amplitude is identical to the carrier's amplitude setting.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E+01, indicating that the AM modulation depth is 50%.

Examples

```
:SOURce1:MOD:AM:DEPTh 50 /*Sets the AM Modulation Depth to 50% for
CH1.*/
:SOURce1:MOD:AM:DEPTh? /*Queries the AM Modulation Depth for CH1.
The query returns 5.0000000000E+01.*/
```

4.3.6.2 [:SOURce[<n>]][:MOD]:AM:DSSC[:STATe]

Syntax

[:SOURce[<*n*>]][:MOD]:AM:DSSC[:STATe] <*state*>

[:SOURce[<*n*>]][:MOD]:AM:DSSC[:STATe]?

Description

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

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<state>	Discrete	{ON|1|OFF|0}	0

Remarks

- DG70000 supports two forms of amplitude modulation, "Normal" and Double Sideband Suppressed Carrier (DSSC). In "normal" AM, the modulated wave consists of the carrier wave and two sidebands. Such a modulation is inefficient because the carrier wave carries no information. If this carrier is suppressed and the saved power is distributed to the two sidebands, then such a process is called Double Sideband Suppressed Carrier (DSSC)
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns 0 or 1.

Examples

```
:SOURce1:MOD:AM:DSSC:STATe ON /*Enables the DSSC mode for CH1.*/
:SOURce1:MOD:AM:DSSC:STATe? /*Queries the on/off state of the DSSC
function for CH1. The query returns 1.*/
```

4.3.6.3 [:SOURce[<n>]][:MOD]:AM:INTernal:FREQuency

Syntax

[:SOURce[<*n*>]][:MOD]:AM:INTernal:FREQuency <*frequency*>

[:SOURce[<*n*>]][:MOD]:AM:INTernal:FREQuency?

Description

Sets or queries the AM Frequency for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	2 mHz to 1 MHz	100 Hz

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+02, indicating that the AM frequency is 150 Hz.

Examples

```
:SOURce1:MOD:AM:INTernal:FREQuency 150 /*Sets the AM Frequency to
150 Hz for CH1.*/
:SOURce1:MOD:AM:INTernal:FREQuency? /*Queries the AM Frequency for
CH1. The query returns 1.5000000000E+02.*/
```

4.3.6.4 [:SOURce[<n>]][:MOD]:AM:INTernal:FUNCtion

Syntax

[:SOURce[<*n*>]][:MOD]:AM:INTernal:FUNCtion <*function*>

[:SOURce[<*n*>]][:MOD]:AM:INTernal:FUNCtion?

Description

Sets or queries the shape of modulation waveform in AM for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<function>	Discrete	{SINusoid|SQUare|TRIangle|RAMP|NRAMp|NOISe}	SINusoid

Remarks

- **SINusoid:** Sine wave; **SQUare:** Square with 50% duty cycle; **TRIangle:** Triangle with 50% symmetry; **RAMP:** UpRamp with 100% symmetry; **NRAMp:** DnRamp with 0% symmetry; **NOISe:** white gaussian noise.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Please use *[:SOURce[<n>]]:WAVE:FUNCtion* to set the carrier wave.

Return Format

The query returns SIN, SQU, TRI, RAMP, NRAM, or NOIS.

Examples

```
:SOURce1:MOD:AM:INTernal:FUNCtion SQUare /*Select a square wave as
the modulation waveform in AM for CH1.*/
:SOURce1:MOD:AM:INTernal:FUNCtion? /*Queries the shape of
modulation waveform in AM for CH1. The query returns SQU.*/
```

4.3.6.5 [:SOURce[<n>]][:MOD]:ASKey:AMPLitude

Syntax

[:SOURce[<*n*>]][:MOD]:ASKey:AMPLitude <*amplitude*>

[:SOURce[<*n*>]][:MOD]:ASKey:AMPLitude?

Description

Sets or queries the ASK Modulation Amplitude for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<amplitude>	Real	25 mVpp to 1 Vpp	700.0 mVpp

Remarks

- In ASK modulation, you can configure the instrument to "shift" its output amplitude between two preset values (called the "carrier amplitude" and the "modulation amplitude").
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-01, indicating that the amplitude is 100 mVpp.

Examples

```
:SOURce1:MOD:ASKey:AMPLitude 0.1 /*Sets the ASK Modulation
Amplitude to 100 mVpp for CH1.*/
:SOURce1:MOD:ASKey:AMPLitude? /*Queries the ASK Modulation
Amplitude for CH1. The query returns 1.0000000000E-01.*/
```

4.3.6.6 [:SOURce[<n>]][:MOD]:ASKey:INTernal[:RATE]

Syntax

[:SOURce[<*n*>]][:MOD]:ASKey:INTernal[:RATE] <*rate*>

[:SOURce[<*n*>]][:MOD]:ASKey:INTernal[:RATE]?

Description

Sets or queries the ASK rate for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<rate>	Real	2 mHz to 1 MHz	100 Hz

Remarks

- ASK rate is the rate at which the output amplitude "shifts" between the carrier amplitude (*[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]*) and modulation amplitude (*[:SOURce[<n>]][:MOD]:ASKey:AMPLitude*).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+02, indicating that the ASK rate is 150 Hz.

Examples

```
:SOURce1:MOD:ASKey:INTernal:RATE 150 /*Sets the ASK rate to 150 Hz
for CH1.*/
:SOURce1:MOD:ASKey:INTernal:RATE? /*Queries the ASK rate for CH1.
The query returns 1.5000000000E+02.*/
```

4.3.6.7 [:SOURce[<n>]][:MOD]:FM:DEViation

Syntax

[:SOURce[<*n*>]][:MOD]:FM:DEViation <*deviation*>

[:SOURce[<*n*>]][:MOD]:FM:DEViation?

Description

Sets or queries the Frequency Deviation in FM for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<deviation>	Real	Refer to *Remarks*	1 kHz

Remarks

- Frequency deviation represents the peak variation in frequency of the modulated waveform from the carrier frequency. Frequency deviation must always be less than or equal to the carrier frequency (*[:SOURce[<n>]]:FREQuency[:FIXed]*). The carrier frequency plus the deviation cannot exceed the selected function's maximum frequency plus 1 kHz. The carrier frequency depends on the selected carrier wave type.
  - Sine: 1 μHz to 1 GHz
  - Square: 1 μHz to 500 MHz
  - Ramp: 1 μHz to 50 MHz
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E+02, indicating that the frequency deviation is 100 Hz.

Examples

```
:SOURce1:MOD:FM:DEViation 100 /*Sets the Frequency Deviation in FM
to 100 Hz for CH1.*/
:SOURce1:MOD:FM:DEViation? /*Queries the Frequency Deviation in FM
for CH1. The query returns 1.0000000000E+02.*/
```

4.3.6.8 [:SOURce[<n>]][:MOD]:FM:INTernal:FREQuency

Syntax

[:SOURce[<*n*>]][:MOD]:FM:INTernal:FREQuency <*frequency*>

[:SOURce[<*n*>]][:MOD]:FM:INTernal:FREQuency?

Description

Sets or queries the FM Frequency for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	2 mHz to 1 MHz	100 Hz

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+02, indicating that the FM frequency is 150 Hz.

Examples

```
:SOURce1:MOD:FM:INTernal:FREQuency 150 /*Sets the FM Frequency to
150 Hz for CH1.*/
:SOURce1:MOD:FM:INTernal:FREQuency? /*Queries the FM Frequency for
CH1. The query returns 1.5000000000E+02.*/
```

4.3.6.9 [:SOURce[<n>]][:MOD]:FM:INTernal:FUNCtion

Syntax

[:SOURce[<*n*>]][:MOD]:FM:INTernal:FUNCtion <*function*>

[:SOURce[<*n*>]][:MOD]:FM:INTernal:FUNCtion?

Description

Sets or queries the shape of modulation waveform in FM for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Name	Type	Range	Default
<function>	Discrete	{SINusoid|SQUare|TRIangle|RAMP|NRAMp|NOISe}	SINusoid

Remarks

- **SINusoid:** Sine wave; **SQUare:** Square with 50% duty cycle; **TRIangle:** Triangle with 50% symmetry; **RAMP:** UpRamp with 100% symmetry; **NRAMp:** DnRamp with 0% symmetry; **NOISe:** white gaussian noise.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Please use *[:SOURce[<n>]]:WAVE:FUNCtion* to set the carrier wave.

Return Format

The query returns SIN, SQU, TRI, RAMP, NRAM, or NOIS.

Examples

```
:SOURce1:MOD:FM:INTernal:FUNCtion SQUare /*Select a square wave as
the modulation waveform in FM for CH1.*/
:SOURce1:MOD:FM:INTernal:FUNCtion? /*Queries the shape of
modulation waveform in FM for CH1. The query returns SQU.*/
```

4.3.6.10 [:SOURce[<n>]][:MOD]:FSKey:FREQuency

Syntax

[:SOURce[<*n*>]][:MOD]:FSKey:FREQuency <*frequency*>

[:SOURce[<*n*>]][:MOD]:FSKey:FREQuency?

Description

Sets or queries the FSK Hop Frequency for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	Refer to *Remarks*	10 kHz

Remarks

- In Frequency Shift Keying (FSK), the generator "shift" its output frequency between the "carrier frequency" (*[:SOURce[<n>]]:FREQuency[:FIXed]*) and the "hop frequency".
- The available hop frequency depends on the selected carrier wave.

  - Sine: 1 μHz to 1 GHz
  - Square: 1 μHz to 500 MHz
  - Ramp: 1 μHz to 50 MHz
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E+03, indicating that the FSK hop frequency is 5 kHz.

Examples

```
:SOURce1:MOD:FSKey:FREQuency 5000 /*Sets the FSK Hop Frequency to
5 kHz for CH1.*/
:SOURce1:MOD:FSKey:FREQuency? /*Queries the FSK Hop Frequency for
CH1. The query returns 5.0000000000E+03.*/
```

4.3.6.11 [:SOURce[<n>]][:MOD]:FSKey:INTernal[:RATE]

Syntax

[:SOURce[<*n*>]][:MOD]:FSKey:INTernal[:RATE] <*rate*>

[:SOURce[<*n*>]][:MOD]:FSKey:INTernal[:RATE]?

Description

Sets or queries the FSK rate for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<rate>	Real	2 mHz to 1 MHz	100 Hz

Remarks

- FSK rate is the rate at which the output frequency "shifts" between the carrier frequency (*[:SOURce[<n>]]:FREQuency[:FIXed]*) and the hop frequency (*[:SOURce[<n>]][:MOD]:FSKey:FREQuency*).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+02, indicating that the FSK rate is 150 Hz.

Examples

```
:SOURce1:MOD:FSKey:INTernal:RATE 150 /*Sets the FSK rate to 150 Hz
for CH1.*/
:SOURce1:MOD:FSKey:INTernal:RATE? /*Queries the FSK rate for CH1.
The query returns 1.5000000000E+02.*/
```

4.3.6.12 [:SOURce[<n>]][:MOD]:PM:DEViation

Syntax

[:SOURce[<*n*>]][:MOD]:PM:DEViation <*deviation*>

[:SOURce[<*n*>]][:MOD]:PM:DEViation?

Description

Sets or queries the Phase Deviation in PM mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<deviation>	Real	0° to 360°	90°

Remarks

- The phase deviation represents the peak variation in phase of the modulated waveform from the carrier waveform.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E+01, indicating that the phase deviation is 50°.

Examples

```
:SOURce1:MOD:PM:DEViation 50 /*Sets the Phase Deviation in PM mode
to 50° for CH1.*/
:SOURce1:MOD:PM:DEViation? /*Queries the Phase Deviation in PM
mode for CH1. The query returns 5.0000000000E+01.*/
```

4.3.6.13 [:SOURce[<n>]][:MOD]:PM:INTernal:FREQuency

Syntax

[:SOURce[<*n*>]][:MOD]:PM:INTernal:FREQuency <*frequency*>

[:SOURce[<*n*>]][:MOD]:PM:INTernal:FREQuency?

Description

Sets or queries the PM Frequency for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<frequency>	Real	2 mHz to 1 MHz	100 Hz

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+02, indicating that the PM frequency is 150 Hz.

Examples

```
:SOURce1:MOD:PM:INTernal:FREQuency 150 /*Sets the PM Frequency to
150 Hz for CH1.*/
:SOURce1:MOD:PM:INTernal:FREQuency? /*Queries the PM Frequency for
CH1. The query returns 1.5000000000E+02.*/
```

4.3.6.14 [:SOURce[<n>]][:MOD]:PM:INTernal:FUNCtion

Syntax

[:SOURce[<*n*>]][:MOD]:PM:INTernal:FUNCtion <*function*>

[:SOURce[<*n*>]][:MOD]:PM:INTernal:FUNCtion?

Description

Sets or queries the shape of modulation waveform in PM for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<function>	Discrete	{SINusoid|SQUare|TRIangle|RAMP|NRAMp|NOISe}	SINusoid

Remarks

- **SINusoid:** Sine wave; **SQUare:** Square with 50% duty cycle; **TRIangle:** Triangle with 50% symmetry; **RAMP:** UpRamp with 100% symmetry; **NRAMp:** DnRamp with 0% symmetry; **NOISe:** white gaussian noise.

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Please use *[:SOURce[<n>]]:WAVE:FUNCtion* to set the carrier wave.

Return Format

The query returns SIN, SQU, TRI, RAMP, NRAM, or NOIS.

Examples

```
:SOURce1:MOD:PM:INTernal:FUNCtion SQUare /*Select a square wave as
the modulation waveform in PM for CH1.*/
:SOURce1:MOD:PM:INTernal:FUNCtion? /*Queries the shape of
modulation waveform in PM for CH1. The query returns SQU.*/
```

4.3.6.15 [:SOURce[<n>]][:MOD]:PSKey:INTernal[:RATE]

Syntax

[:SOURce[<*n*>]][:MOD]:PSKey:INTernal[:RATE] <*rate*>

[:SOURce[<*n*>]][:MOD]:PSKey:INTernal[:RATE]?

Description

Sets or queries the PSK rate for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<rate>	Real	2 mHz to 1 MHz	100 Hz

Remarks

- PSK rate is the rate at which the output phase "shifts" between the carrier phase (*[:SOURce[<n>]]:PHASe[:ADJust]*) and modulation phase (*[:SOURce[<n>]][:MOD]:PSKey:PHASe*).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.5000000000E+02, indicating that the PSK rate is 150 Hz.

Examples

```
:SOURce1:MOD:PSKey:INTernal:RATE 150 /*Sets the PSK rate to 150 Hz
for CH1.*/
:SOURce1:MOD:PSKey:INTernal:RATE? /*Queries the PSK rate for CH1.
The query returns 1.5000000000E+02.*/
```

4.3.6.16 [:SOURce[<n>]][:MOD]:PSKey:PHASe

Syntax

[:SOURce[<*n*>]][:MOD]:PSKey:PHASe <*phase*>

[:SOURce[<*n*>]][:MOD]:PSKey:PHASe?

Description

Sets or queries the PSK phase for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<phase>	Real	0° to 360°	180°

Remarks

- In Phase Shift Keying (PSK), the generator "shifts" its output phase between two phase settings, carrier phase (*[:SOURce[<n>]]:PHASe[:ADJust]*) and modulation phase.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E+01, indicating that the PSK phase is 50°.

Examples

```
:SOURce1:MOD:PSKey:PHASe 50 /*Sets the PSK phase to 50° for CH1.*/
:SOURce1:MOD:PSKey:PHASe? /*Queries the PSK phase for CH1. The
query returns 5.0000000000E+01.*/
```

4.3.6.17 [:SOURce[<n>]][:MOD]:TYPE

Syntax

[:SOURce[<*n*>]][:MOD]:TYPE <*type*>

[:SOURce[<*n*>]][:MOD]:TYPE?

Description

Sets or queries the modulation type for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<type>	Discrete	{AM|FM|PM|ASK|FSK|PSK}	AM

Remarks

- When the output mode (*[:SOURce[<n>]]:MODE:FUNCtion*) is set to Modulation (MODulation), you can use this command to set the modulation type.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns AM, FM, PM, ASK, FSK, or PSK.

Examples

```
:SOURce1:MOD:TYPE PM /*Sets the modulation type to PM for CH1.*/
:SOURce1:MOD:TYPE? /*Queries the modulation type for CH1. The
query returns PM.*/
```

4.3.7 [:SOURce[<n>]]:PERiod[:FIXed]

Syntax

[:SOURce[<*n*>]]:PERiod[:FIXed] <*period*>

[:SOURce[<*n*>]]:PERiod[:FIXed]?

Description

Sets or queries the Period of basic wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<period>	Real	Refer to *Remarks*	1 ms

Remarks

- For the period available for different waves, please refer to *Table 4.8 The frequency and period range for basic wave* .
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

- When the wave type is changed, the instrument still uses the period if the period is valid. Otherwise, the instrument automatically sets the period as the lower limit for the new wave type.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-01, indicating that the period is 0.1 s.

Examples

```
:SOURce1:PERiod:FIXed 0.1 /*Sets the Period of basic wave to 0.1 s
for CH1.*/
:SOURce1:PERiod:FIXed? /*Queries the Period of basic wave for CH1.
The query returns 1.0000000000E-01.*/
```

4.3.8 [:SOURce[<n>]]:PHASe[:ADJust]

Syntax

[:SOURce[<*n*>]]:PHASe[:ADJust] <*phase*>

[:SOURce[<*n*>]]:PHASe[:ADJust]?

Description

Sets or queries the Starting Phase for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<phase>	Real	0° to 360°	0°

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E+01, indicating that the starting phase is 10°.

Examples

```
:SOURce1:PHASe:ADJust 10 /*Sets the Starting Phase to 10° for CH1.*/
:SOURce1:PHASe:ADJust? /*Queries the Starting Phase for CH1. The
query returns 1.0000000000E+01.*/
```

4.3.9 [:SOURce[<n>]]:PULSe:WIDTh

Syntax

[:SOURce[<*n*>]]:PULSe:WIDTh <*seconds*>

[:SOURce[<*n*>]]:PULSe:WIDTh?

Description

Sets or queries the Pulse Width of Pulse wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<seconds>	Real	Refer to *Remarks*	500 µs

Remarks

- The pulse width must conform to the following restrictions determined by the period (*[:SOURce[<n>]]:PERiod[:FIXed]*) and the minimum pulse width (Wmin: 640 ps).
  - Pulse Width ≥ Minimum Pulse Width. When the period is greater than 1.64 µs, Pulse Width ≥ Pulse Period/2560.
  - Pulse Width ≤ Period-Minimum Pulse Width
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-01, indicating that the pulse width is 0.1 s.

Examples

```
:SOURce1:PULSe:WIDTh 0.1 /*Sets the Pulse Width of Pulse wave to
0.1 s for CH1.*/
:SOURce1:PULSe:WIDTh? /*Queries the Pulse Width of Pulse wave for
CH1. The query returns 1.0000000000E-01.*/
```

4.3.10 [:SOURce[<n>]]:PULSe:TRANsition:TRAiling

Syntax

[:SOURce[<*n*>]]:PULSe:TRANsition:TRAiling <*seconds*>

[:SOURce[<*n*>]]:PULSe:TRANsition:TRAiling?

Description

Sets or queries the falling edge time of Pulse wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<seconds>	Real	300 ps to 1 s	300 ps

Remarks

- The falling edge time is the transition time for the falling edge of a pulse from 90% to 10% threshold.
- The Falling Edge Time is affected by the Pulse Width ([:SOURce[<n>]]:PULSe:WIDTh): Falling Edge Time ≤ 0.625×Pulse Width. If the value is out of range, the instrument will adjust the edge time to accommodate the specified pulse width.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 3.5000000000E-08, indicating that the falling edge time is 35 ns.

Examples

```
:SOURce1:PULSe:TRANsition:TRAiling 0.000000035 /*Sets the falling
edge time of Pulse wave to 35 ns for CH1.*/
:SOURce1:PULSe:TRANsition:TRAiling? /*Queries the falling edge time
of Pulse wave for CH1. The query returns 3.5000000000E-08.*/
```

4.3.11 [:SOURce[<n>]]:PULSe:TRANsition[:LEADing]

Syntax

[:SOURce[<*n*>]]:PULSe:TRANsition[:LEADing] <*seconds*>

[:SOURce[<*n*>]]:PULSe:TRANsition[:LEADing]?

Description

Sets or queries the rising edge time of Pulse wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1

Name	Type	Range	Default
<seconds>	Real	300 ps to 1 s	300 ps

Remarks

- The rising edge time is the transition time for the rising edge of a pulse from 10% to 90% threshold.
- The Rising Edge Time is affected by the Pulse Width (*[:SOURce[<n>]]:PULSe:WIDTh*): Rising Edge Time ≤ 0.625×Pulse Width. If the value is out of range, the instrument will adjust the edge time to accommodate the specified pulse width.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 3.5000000000E-08, indicating that the rising edge time is 35 ns.

Examples

```
:SOURce1:PULSe:TRANsition:LEADing 0.000000035 /*Sets the rising
edge time of Pulse wave to 35 ns for CH1.*/
:SOURce1:PULSe:TRANsition:LEADing? /*Queries the rising edge time
for CH1. The query returns 3.5000000000E-08.*/
```

4.3.12 [:SOURce[<n>]]:PULSe:DCYCle

Syntax

[:SOURce[<*n*>]]:PULSe:DCYCle <*percent*>

[:SOURce[<*n*>]]:PULSe:DCYCle?

Description

Sets or queries the Duty Cycle of Pulse wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<percent>	Real	Refer to *Remarks*	50%

Remarks

- The pulse duty cycle is defined as the percentage of the pulse width to the pulse period (*[:SOURce[<n>]]:PERiod[:FIXed]*). The pulse duty cycle and pulse width are interdependent. Changing one of them automatically modifies the other.
- The duty cycle must conform to the following restrictions determined by the "Minimum Pulse Width" (640 ps) and the "Period":

  (Minimum Pulse Width/Period)*100% ≤ Duty Cycle ≤ (1-Minimum Pulse Width/Period)*100%
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 4.5000000000E+01, indicating that the duty cycle is 45%.

Examples

```
:SOURce1:PULSe:DCYCle 45 /*Sets the Duty Cycle of Pulse wave to 45%
for CH1.*/
:SOURce1:PULSe:DCYCle? /*Queries the Duty Cycle of Pulse wave for
CH1. The query returns 4.5000000000E+01.*/
```

4.3.13 [:SOURce[<n>]]:SWEep

4.3.13.1 [:SOURce[<n>]]:SWEep:SPACing

Syntax

[:SOURce[<*n*>]]:SWEep:SPACing <*type*>

[:SOURce[<*n*>]]:SWEep:SPACing?

Description

Sets or queries the sweep type for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<type>	Discrete	{LINear|LOGarithmic|STEP}	LINear

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

When the output mode (*[:SOURce[<n>]]:MODE:FUNCtion*) is set to Sweep (SWEep), you can use this command to set the sweep type. DG70000 provides the following three sweep types.

- **LINear:** Linear Sweep. The instrument varies the output frequency linearly during the sweep, changing the output frequency by a constant number of Hz per second. It is characterized by "Start Frequency" (*[:SOURce[<n>]]:FREQuency:STARt*), "Stop Frequency" (*[:SOURce[<n>]]:FREQuency:STOP*), and "Sweep Time" (*[:SOURce[<n>]]:SWEep:TIME*).
- **LOGarithmic:** Logarithmic Sweep. The instrument varies the output frequency logarithmically, changing the frequency by a constant number of octaves or decades per second. It is characterized by "Start Frequency", "Stop Frequency", and "Sweep Time".
- **STEP:** Step Sweep. The instrument "steps" through a list of frequencies. The period that the output signal dwells on each frequency is determined by "Sweep Time" and "Step" (*[:SOURce[<n>]]:SWEep:STEP*).

Return Format

The query returns LIN, LOG, or STEP.

Examples

```
:SOURce1:SWEep:SPACing LINear /*Sets the sweep type to linear
sweep for CH1.*/
:SOURce1:SWEep:SPACing? /*Queries the sweep type for CH1. The
query returns LIN.*/
```

4.3.13.2 [:SOURce[<n>]]:SWEep:TIME

Syntax

[:SOURce[<*n*>]]:SWEep:TIME <*time*>

[:SOURce[<*n*>]]:SWEep:TIME?

Description

Sets or queries the Sweep Time for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<time>	Real	1 ms to 500 s	1 s

Remarks

- Sweep time specifies the number of seconds required to sweep from the "Start Frequency" ([:SOURce[<n>]]:FREQuency:STARt) to the "Stop Frequency" ([:SOURce[<n>]]:FREQuency:STOP).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Modifying the "Sweep Time" enables the instrument to sweep from the specified "Start Frequency".

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E+00, indicating that the sweep time is 5 s.

Examples

```
:SOURce1:SWEep:TIME 5 /*Sets the Sweep Time to 5 s for CH1.*/
:SOURce1:SWEep:TIME? /*Queries the Sweep Time  for CH1. The query
returns 5.0000000000E+00.*/
```

4.3.13.3 [:SOURce[<n>]]:SWEep:RTIMe

Syntax

[:SOURce[<*n*>]]:SWEep:RTIMe <*time*>

[:SOURce[<*n*>]]:SWEep:RTIMe?

Description

Sets or queries the Return Time in Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<time>	Real	Refer to *Remarks*	0 s

Remarks

- <time> is 0 s or ranges from 100 μs to 500 s. A <time> setting value less than 100 μs will be automatically set to 0 s.
- Return time specifies the number of seconds to return from the "Stop Frequency" ([:SOURce[<n>]]:FREQuency:STOP) to the "Start Frequency" ([:SOURce[<n>]]:FREQuency:STARt).
- If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

- Modifying the "Return Time" enables the instrument to sweep from the specified "Start Frequency".

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-03, indicating that the return time is 1 ms.

Examples

```
:SOURce1:SWEep:RTIMe 0.001 /*Sets the Return Time to 1 ms for
CH1.*/
:SOURce1:SWEep:RTIMe? /*Queries the Return Time in Sweep mode for
CH1. The query returns 1.0000000000E-03.*/
```

4.3.13.4 [:SOURce[<n>]]:SWEep:HTIMe:STARt

Syntax

[:SOURce[<*n*>]]:SWEep:HTIMe:STARt <*time*>

[:SOURce[<*n*>]]:SWEep:HTIMe:STARt?

Description

Sets or queries the Start Hold time in Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<time>	Real	Refer to *Remarks*	0 ns

Remarks

- <time> is 0 s or ranges from 100 μs to 500 s. A <time> setting value less than 100 μs will be automatically set to 0 s.
- Start hold time specifies the number of seconds for the sweep to remain at the "Start Frequency" (*[:SOURce[<n>]]:FREQuency:STARt*). After the start hold time expires, the generator continues to sweep at varied frequencies according to the current sweep type.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Modifying the "Start Hold" time enables the instrument to sweep from the specified "Start Frequency".

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-03, indicating that the start hold time is 1 ms.

Examples

```
:SOURce1:SWEep:HTIMe:STARt 0.001 /*Sets the Start Hold time to 1
ms for CH1.*/
:SOURce1:SWEep:HTIMe:STARt? /*Queries the Start Hold time in Sweep
mode for CH1. The query returns 1.0000000000E-03.*/
```

4.3.13.5 [:SOURce[<n>]]:SWEep:HTIMe:STOP

Syntax

[:SOURce[<*n*>]]:SWEep:HTIMe:STOP <*time*>

[:SOURce[<*n*>]]:SWEep:HTIMe:STOP?

Description

Sets or queries the Stop Hold time in Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<time>	Real	Refer to *Remarks*	0 s

Remarks

- <time> is 0 s or ranges from 100 μs to 500 s. A <time> setting value less than 100 μs will be automatically set to 0 s.
- Stop hold time specifies the number of seconds for the sweep to remain at the "Stop Frequency" after the generator sweeps from the "Start Frequency" (*[:SOURce[<n>]]:FREQuency:STARt*) to the "Stop Frequency" (*[:SOURce[<n>]]:FREQuency:STOP*).
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.
- Modifying the "Stop Hold" time enables the instrument to sweep from the specified "Start Frequency".

Return Format

The query returns the value in scientific notation. For example, the query might return 1.0000000000E-03, indicating that the stop hold time is 1 ms.

Examples

```
:SOURce1:SWEep:HTIMe:STOP 0.001 /*Sets the Stop Hold time to 1 ms
for CH1.*/
:SOURce1:SWEep:HTIMe:STOP? /*Queries the Stop Hold time in Sweep
mode for CH1. The query returns 1.0000000000E-03.*/
```

4.3.13.6 [:SOURce[<n>]]:SWEep:STEP

Syntax

[:SOURce[<*n*>]]:SWEep:STEP <*step*>

[:SOURce[<*n*>]]:SWEep:STEP?

Description

Sets or queries the number of steps in Step Sweep mode for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<step>	Integer	2 to 1024	2

Remarks

- The number of steps specifies the steps required for the signal to sweep between "Start Frequency" (*[:SOURce[<n>]]:FREQuency:STARt*) and "Stop Frequency" (*[:SOURce[<n>]]:FREQuency:STOP*). This command is available only when the sweep type is set to step sweep.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns an integer between 2 and 1024. For example, the query might return 5.

Examples

```
:SOURce1:SWEep:STEP 5 /*Sets the number of steps in Step Sweep
mode to 5 for CH1.*/
:SOURce1:SWEep:STEP? /*Queries the number of steps in Step Sweep
mode for CH1. The query returns 5.*/
```

4.3.13.7 [:SOURce[<n>]]:SWEep:TRIGger:SOURce

Syntax

[:SOURce[<*n*>]]:SWEep:TRIGger:SOURce <*source*>

[:SOURce[<*n*>]]:SWEep:TRIGger:SOURce?

Description

Sets or queries the trigger source for the specified channel in Sweep mode.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<source>	Discrete	{INTernal|MANual}	INTernal

Remarks

- **INTernal:** Internal trigger source. The generator outputs a continuous sweep. The trigger period is determined by the specified sweep time (*[:SOURce[n]]:SWEep:TIME*), return time (*[:SOURce[n]]:SWEep:RTIMe*), start hold time (*[:SOURce[n]]:SWEep:HTIMe:STARt*), and stop hold time (*[:SOURce[n]]:SWEep:HTIMe[:STOP]*).
- **MANual:** Manual trigger. The instrument outputs one sweep each time you click or tap the manual trigger button or send the trigger command. The trigger command is valid only when the specified channel output is enabled.
- If [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns INT or MAN.

Examples

```
:SOURce1:SWEep:TRIGger:SOURce MANual /*Sets the sweep trigger
source to Manual for CH1.*/
:SOURce1:SWEep:TRIGger:SOURce? /*Queries the sweep trigger source
for CH1. The query returns MAN.*/
```

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

Syntax

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

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

Description

Sets or queries the Amplitude of basic wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<amplitude>	Real	25.0 mVpp to 1.0000 Vpp	500 mVpp

Remarks

- You can also use the "High Level" ([:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:HIGH) and "Low Level" ([:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:LOW) to set the Amplitude and Offset: Amplitude=High Level-Low Level; Offset=(High Level+Low Level)/2.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 5.0000000000E-01, indicating that the amplitude is 500 mVpp.

Examples

```
:SOURce1:VOLTage:LEVel:IMMediate:AMPLitude 0.5 /*Sets the Amplitude
of basic wave to 500 mVpp for CH1.*/
:SOURce1:VOLTage:LEVel:IMMediate:AMPLitude? /*Queries the Amplitude
of basic wave for CH1. The query returns 5.0000000000E-01.*/
```

4.3.15 [:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:HIGH

Syntax

[:SOURce[<*n*>]]:VOLTage[:LEVel][:IMMediate]:HIGH <*voltage*>

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

Description

Sets or queries the High Level of basic wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<voltage>	Real	-987.5 mV to 1.5 V	250 mV

Remarks

- The High Level is determined by the current Low Level (*[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:LOW*): High Level ≥ Low Level+25 mV.
- You can also use "Amplitude" (*[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]*) and "Offset" (*[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:OFFSet*) to set the High and Low Level: High Level=Offset+Amplitude/2; Low Level=Offset-Amplitude/2.
- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 3.5000000000E-01, indicating that the high level is 350 mV.

Examples

```
:SOURce1:VOLTage:LEVel:IMMediate:HIGH 0.35 /*Sets the High Level to
350 mV for CH1.*/
:SOURce1:VOLTage:LEVel:IMMediate:HIGH? /*Queries the High Level for
CH1. The query returns 3.5000000000E-01.*/
```

4.3.16 [:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:LOW

Syntax

[:SOURce[<*n*>]]:VOLTage[:LEVel][:IMMediate]:LOW <*voltage*>

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

Description

Sets or queries the Low Level of basic wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<voltage>	Real	-1.5 V to 987.5 mV	-250 mV

Remarks

- The Low Level is determined by the current High Level (*[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:HIGH*): Low Level ≤ High Level-25 mV.
- You can also use "Amplitude" (*[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]*) and "Offset" (*[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:OFFSet*)

to set the High and Low Level: High Level=Offset+Amplitude/2; Low Level=Offset-Amplitude/2.

- When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return -3.5000000000E-01, indicating that the low level is -350 mV.

Examples

```
:SOURce1:VOLTage:LEVel:IMMediate:LOW -0.35 /*Sets the Low Level to
-350 mV for CH1.*/
:SOURce1:VOLTage:LEVel:IMMediate:LOW? /*Queries the Low Level for
CH1. The query returns -3.5000000000E-01.*/
```

4.3.17 [:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:OFFSet

Syntax

[:SOURce[<*n*>]]:VOLTage[:LEVel][:IMMediate]:OFFSet <*offset*>

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

Description

Sets or queries the Offset of basic wave for the specified channel.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<offset>	Real	-1 V to 1 V	0 mV

Remarks

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns the value in scientific notation. For example, the query might return 2.0000000000E-01, indicating that the offset voltage is 200 mV.

Examples

```
:SOURce1:VOLTage:LEVel:IMMediate:OFFSet 0.2 /*Sets the Offset to
200 mV for CH1.*/
:SOURce1:VOLTage:LEVel:IMMediate:OFFSet? /*Queries the waveform
Offset for CH1. The query returns 2.0000000000E-01.*/
```

4.3.18 [:SOURce[<n>]]:WAVE:FUNCtion

Syntax

[:SOURce[<*n*>]]:WAVE:FUNCtion <*func*>

[:SOURce[<*n*>]]:WAVE:FUNCtion?

Description

Sets or queries the selected wave type in AFG mode.

Parameter

Name	Type	Range	Default
<n>	Discrete	{1|2|3|4}	1
<func>	Discrete	{SINusiod|SQUare|RAMP|PULSe|NOISe}	SINusiod

Remarks

- **SINusiod** selects sine waveform.
- **SQUare** selects square waveform.
- **RAMP** selects ramp waveform.
- **PULSe** selects pulse.
- **NOISe** selects noise.

When [:SOURce[<n>]] or [<n>] is omitted, it is interpreted as CH1.

Return Format

The query returns SIN, SQU, RAMP, PULS, or NOIS.

Examples

```
:SOURce1:WAVE:FUNCtion RAMP /*Sets the wave type to Ramp in AFG
mode for CH1.*/
:SOURce1:WAVE:FUNCtion? /*Queries the basic wave type in AFG mode
for CH1. The query returns RAMP.*/
```

5 Programming Examples

This chapter illustrates how to control the instrument by programming in Excel, Matlab, LabVIEW, Visual Basic, and Visual C++. These examples are programmed based on NI-VISA library.

NI-VISA (National Instrument-Virtual Instrument Software Architecture) is an advanced API developed by National Instrument (NI) for communicating with instruments from a computer. It uses many of the same operations to communicate with instruments, regardless of the interface type (e.g. GPIB, USB, LAN/Ethernet).

The instrument connected to it is considered a VISA "resource". The VISA Descriptor ("Resource Name") specifies the exact name and position of the VISA resource. Before programming, please acquire the correct VISA descriptor.

5.1 Programming Preparations

Before programming, you need to prepare the following tasks:

1. Install the Ultra Sigma software to your PC. Please log in to the RIGOL official website (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.
2. In the manual, the instrument communicates with the PC via the USB DEVICE interface. Connect the USB DEVICE interface on the rear panel of the instrument to the PC by using the USB cable. You can also comminute with the PC via the LAN or GPIB inteface.
3. After the instrument is properly connected to the PC, power on the instrument to start it.
4. Then the wizard dialog box is displayed. Please install the USB Test and Measurement Device (IVI)" following the wizard. You can refer to "Remote Control via USB" in chapter "Remote control" in DG70000 User Guide.
5. Obtain the instrument's USB VISA descriptor.

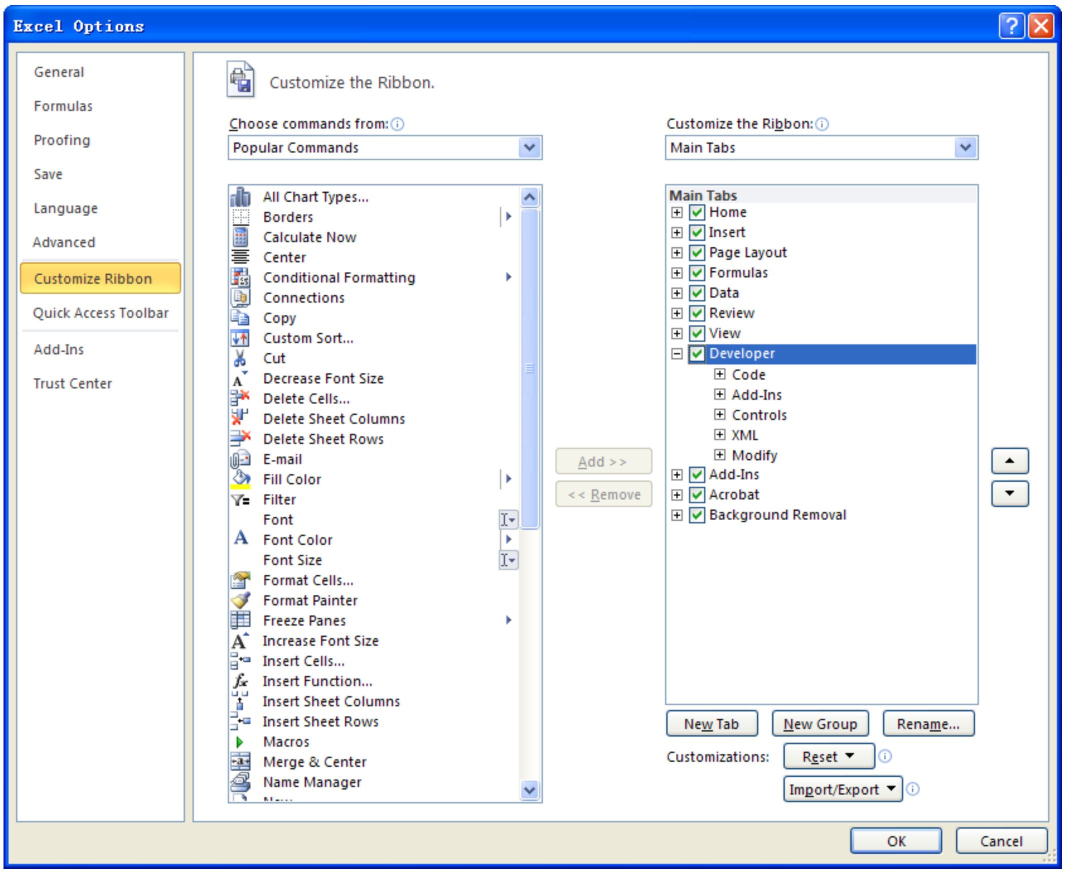
5.2 Excel Programming Example

Program used in this example: Microsoft Excel 2010

Function realized in this example: sending the *IDN? command and reading the instrument information.

1. Open a new Macro-enabled Excel file and name it "Demo_Excel.xlsm" in this example.

2. Run the Demo_Excel.xlsm file. Click **File** > **Options** at the upper-left corner of the Excel file to open the interface as shown in the figure below. Click "Customize Ribbon" at the left, check "Developer" and click **OK**. At this point, the Excel menu bar displays the "Developer" menu.


Excel Options

Customize the Ribbon.

Choose commands from: Popular Commands

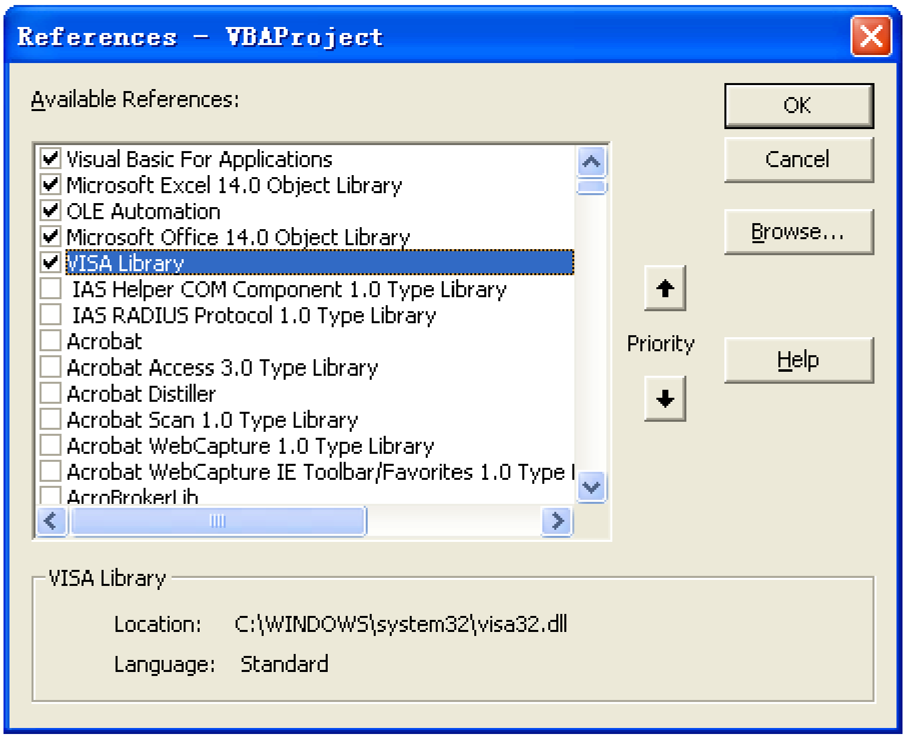
Customize the Ribbon: Main Tabs

Main Tabs: [✓] Home, [✓] Insert, [✓] Page Layout, [✓] Formulas, [✓] Data, [✓] Review, [✓] View, [✓] Developer (Code, Add-Ins, Controls, XML, Modify), [✓] Add-Ins, [✓] Acrobat, [✓] Background Removal

3. Enter a device resource descriptor into a cell of the file as shown in the figure below. For example, the device resource descriptor is USB0::0x1AB1::0x0600::DG7F910400000::INSTR. Input it into SHEET1.CELLS(1,2) (i.g. the B1 cell in Sheet1). Click the Developer menu and select the Visual Basic option to open the Microsoft Visual Basic.

4. Select Tools (T) in the Microsoft Visual Basic menu bar and click References.

   In the displayed dialog box, select VISA Library, and click OK to refer to VISA Library.



- [✓] Visual Basic For Applications
- [✓] Microsoft Excel 14.0 Object Library
- [✓] OLE Automation
- [✓] Microsoft Office 14.0 Object Library
- [✓] VISA Library
- [ ] IAS Helper COM Component 1.0 Type Library
- [ ] IAS RADIUS Protocol 1.0 Type Library
- [ ] Acrobat
- [ ] Acrobat Access 3.0 Type Library
- [ ] Acrobat Distiller
- [ ] Acrobat Scan 1.0 Type Library
- [ ] Acrobat WebCapture 1.0 Type Library
- [ ] Acrobat WebCapture IE Toolbar/Favorites 1.0 Type l
- [ ] AcroBrokerLib

VISA Library

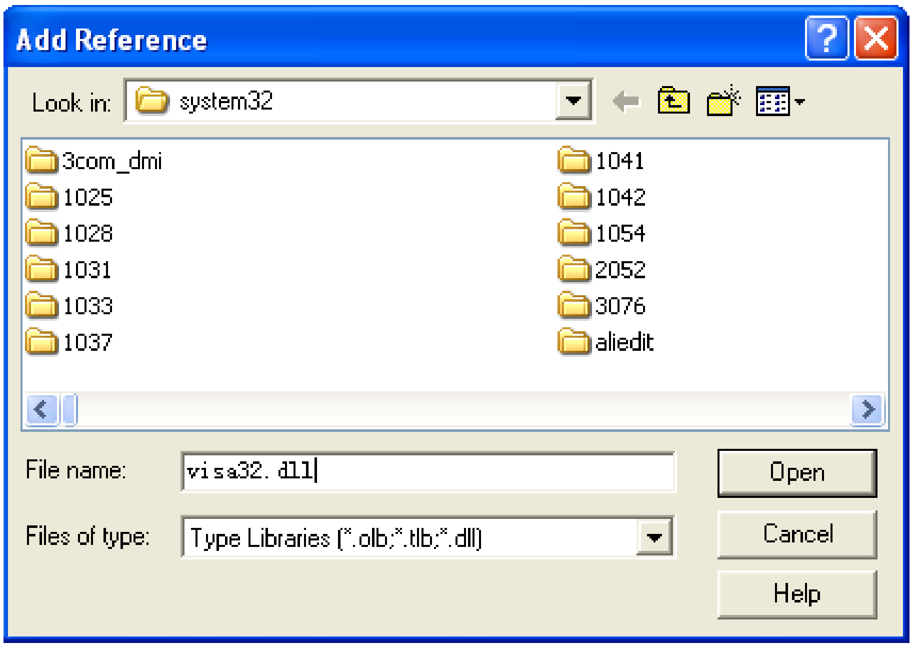
Location: C:\WINDOWS\system32\visa32.dll

Language: Standard

If you cannot find VISA Library in the left section of the above dialog box, please follow the method below to find it.

a. Make sure that your PC has installed the NI-VISA library.

b. Click Browse... at the right section to search visa32.dll from C:\WINDOWS \system32, as shown in the figure below.



File name: visa32.dll

Files of type: Type Libraries (*.olb;*.tlb;*.dll)

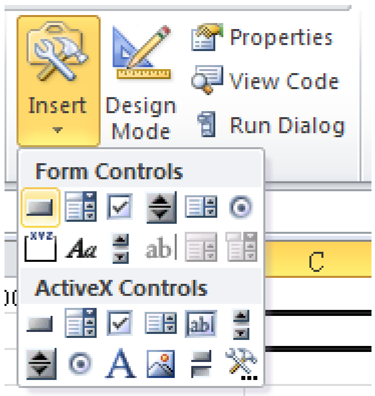
5. Click View Code under Developer menu to enter the interface of Microsoft Visual Basic. Add the following codes and save it.

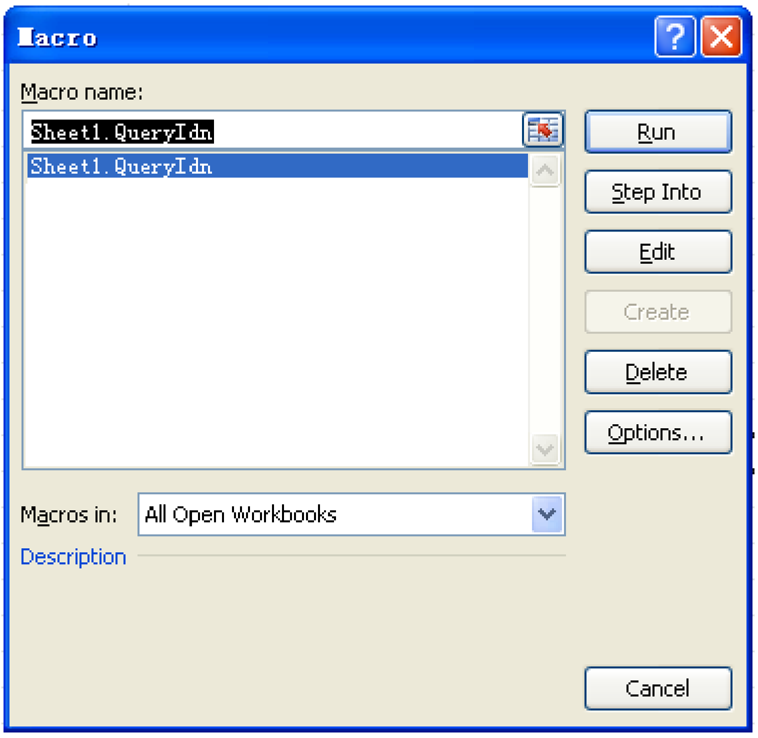
CAUTION

If the Excel file created in Step 1 does not enable the Macros, a prompt message "The following features cannot be saved in macro-free workbooks" will be displayed. In this case, please save the file as a macro-enabled file type (filename with a suffix of ".xlsm").

```
Sub QueryIdn()

    Dim viDefRm As Long
    Dim viDevice As Long
    Dim viErr As Long
    Dim cmdStr As String
    Dim idnStr As String * 128
    Dim ret As Long
    'Turn on the device, and the device resource descriptor is in
CELLS(1,2) of SHEET1'
    viErr = visa.viOpenDefaultRM(viDefRm)
    viErr = visa.viOpen(viDefRm, Sheet1.Cells(1, 2), 0, 5000,
viDevice)

   'Send request, read the data, and the return value is in
CELLS(2,2) of SHEET1'
    cmdStr = "*IDN?"
    viErr = visa.viWrite(viDevice, cmdStr, Len(cmdStr), ret)
    viErr = visa.viRead(viDevice, idnStr, 128, ret)
    Sheet1.Cells(2, 2) = idnStr

   'Turn off the device'
    visa.viClose (viDevice)
    visa.viClose (viDefRm)

End Sub
```

6. Add the button control. Click Insert under the "Developer" menu, and select a button control under the "Form Controls" menu item and put it into the Excel cell. At this time, the "Assign Macro" dialog box is displayed, select "Sheet1.QueryIdn" and click OK.





The default name of the button is "Button1". Right-click the button and select Edit Text in the pop-up menu to change the button name to "*IDN?".

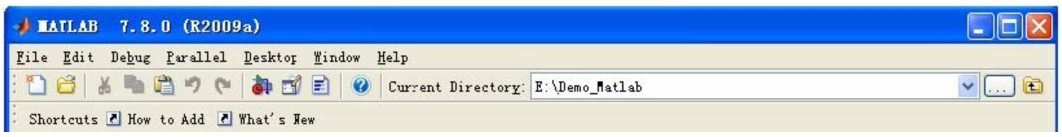
7. Click the "*IDN?" button to send request and read data. The returned value is in CELLS(2,2) of SHEET1.

5.3 Matlab Programming Example

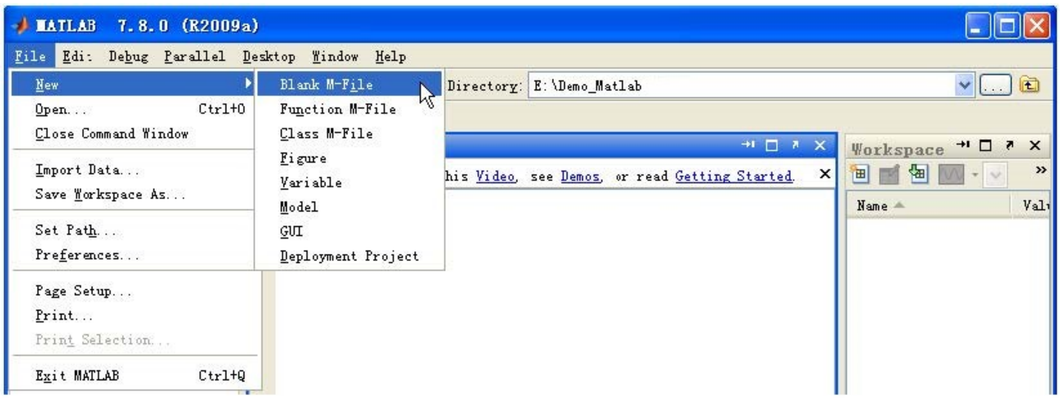
Program used in this example: MATLAB R2009a

Function realized in this example: query the waveform offset for CH1.

1. Run Matlab and modify the current directory in the Current Directory field. The current directory is E:\Demo_Matlab in this example.


MATLAB 7.8.0 (R2009a)
File Edit Debug Parallel Desktop Window Help
Current Directory: E:\Demo_Matlab
Shortcuts How to Add What's New

2. Click **File>New>Blank M-File** in the Matlab interface to create an empty file with the suffix "M".



3. Add the following codes in the file:

```
DG70000 =
visa( 'ni','USB0::0x1AB1::0x0642::DG70000000001::INSTR' );
%Create VISA object
fopen(DG70000); %Open the created VISA object
fprintf(DG70000, ':VOLT:OFFS?' ); %Send requests
query_CH1 = fscanf(DG70000); %Query the data
fclose(DG70000); %Close VISA object
display(query_CH1) %Display the read device information
```

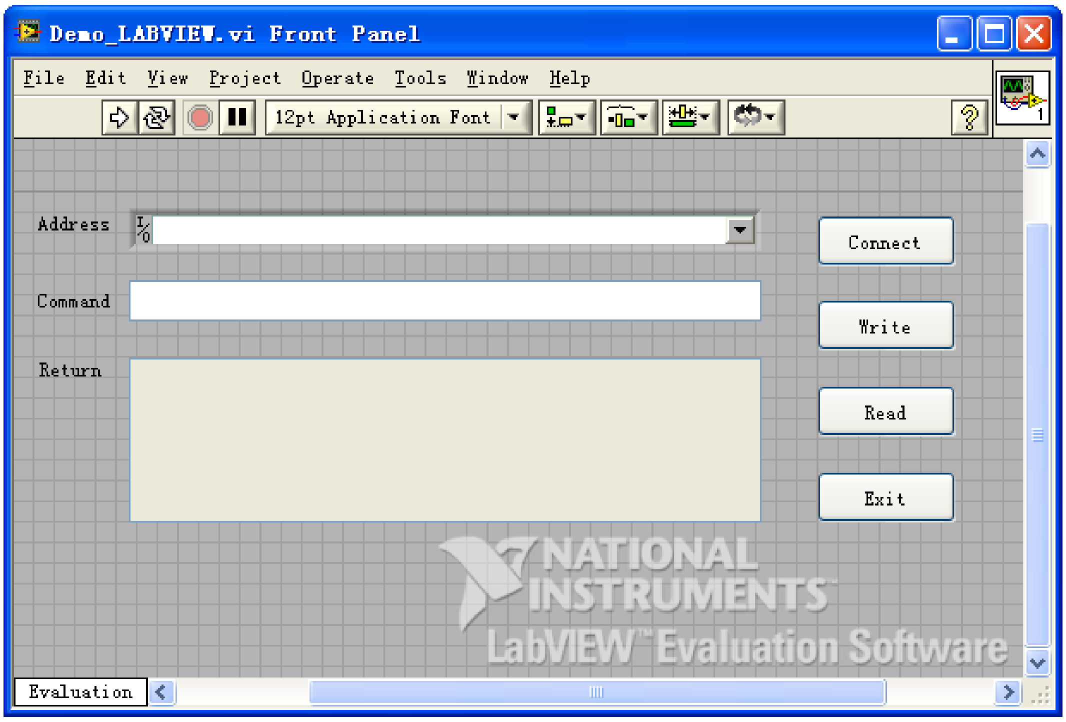
4. Save the file to the current directory. The M file in this example is named DG70000_Demo_MATLAB.m.
5. Run M file and the results will be displayed in the command window.

5.4 LabVIEW Programming Example

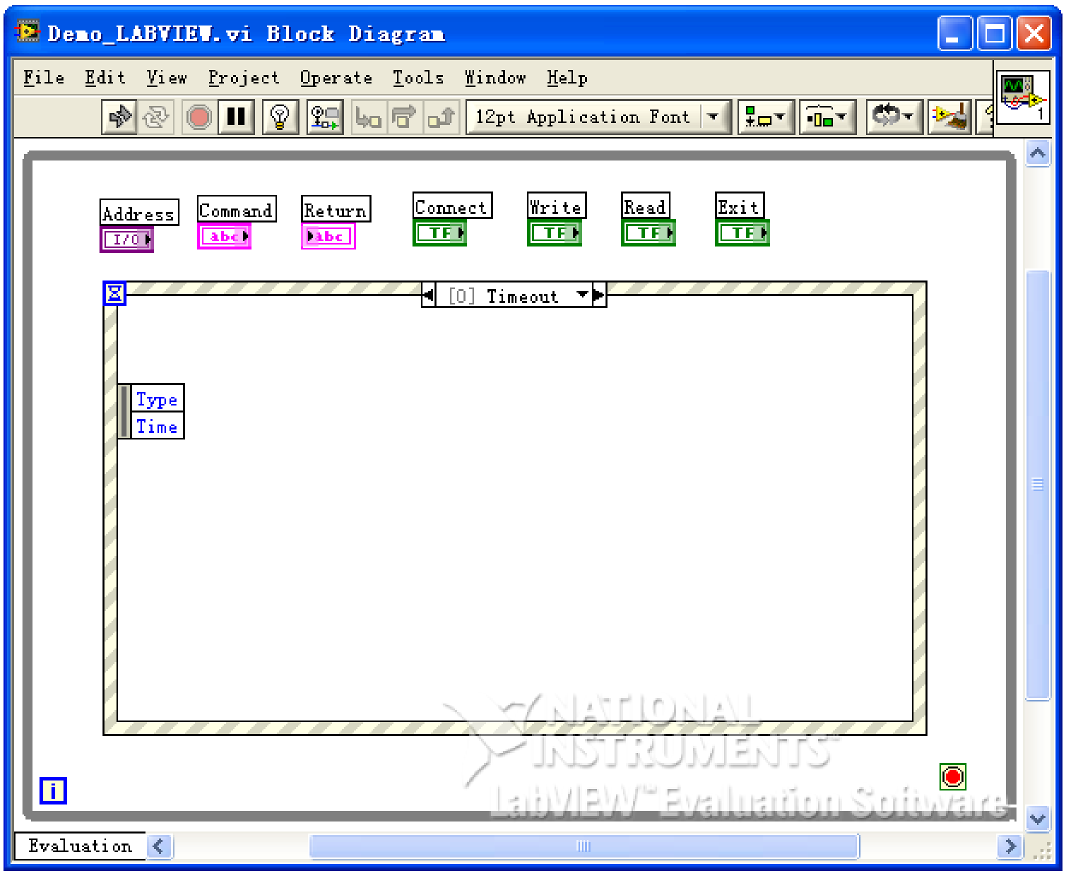
Program used in this example: LabVIEW 2009

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.

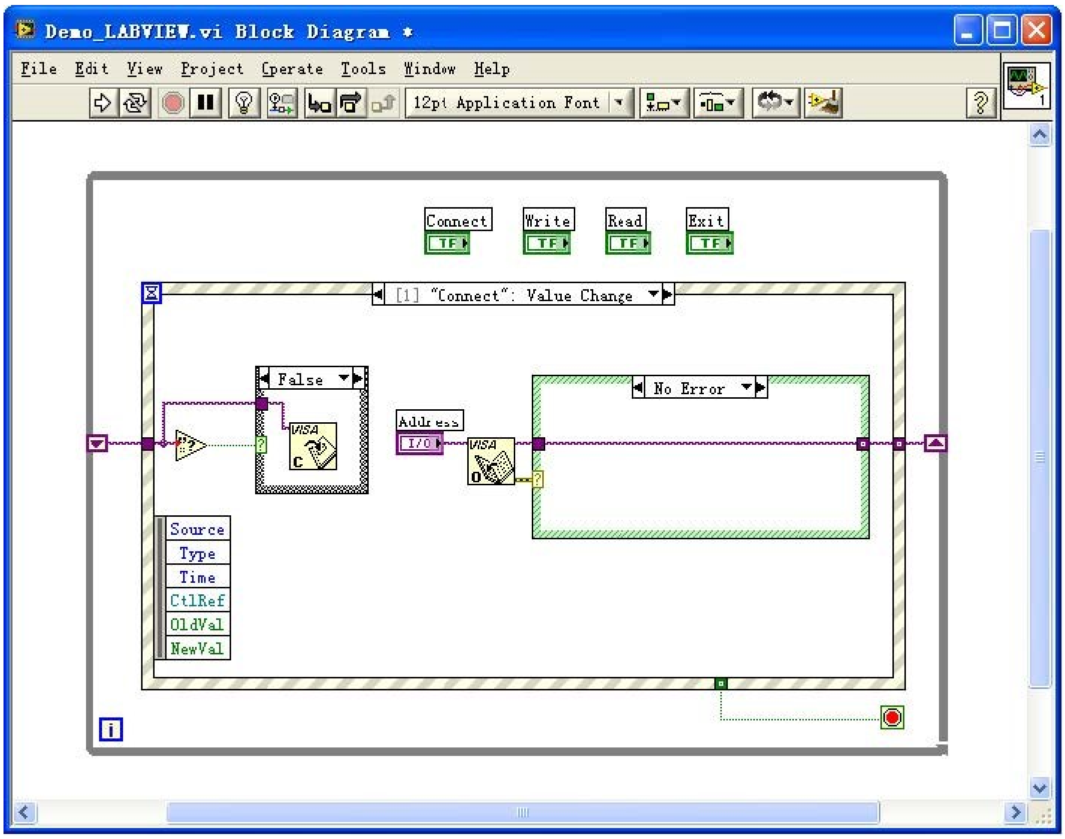

Demo_LABVIEW.vi Front Panel
File Edit View Project Operate Tools Window Help
12pt Application Font
Address
Command
Return
Connect
Write
Read
Exit
NATIONAL INSTRUMENTS LabVIEW Evaluation Software
Evaluation

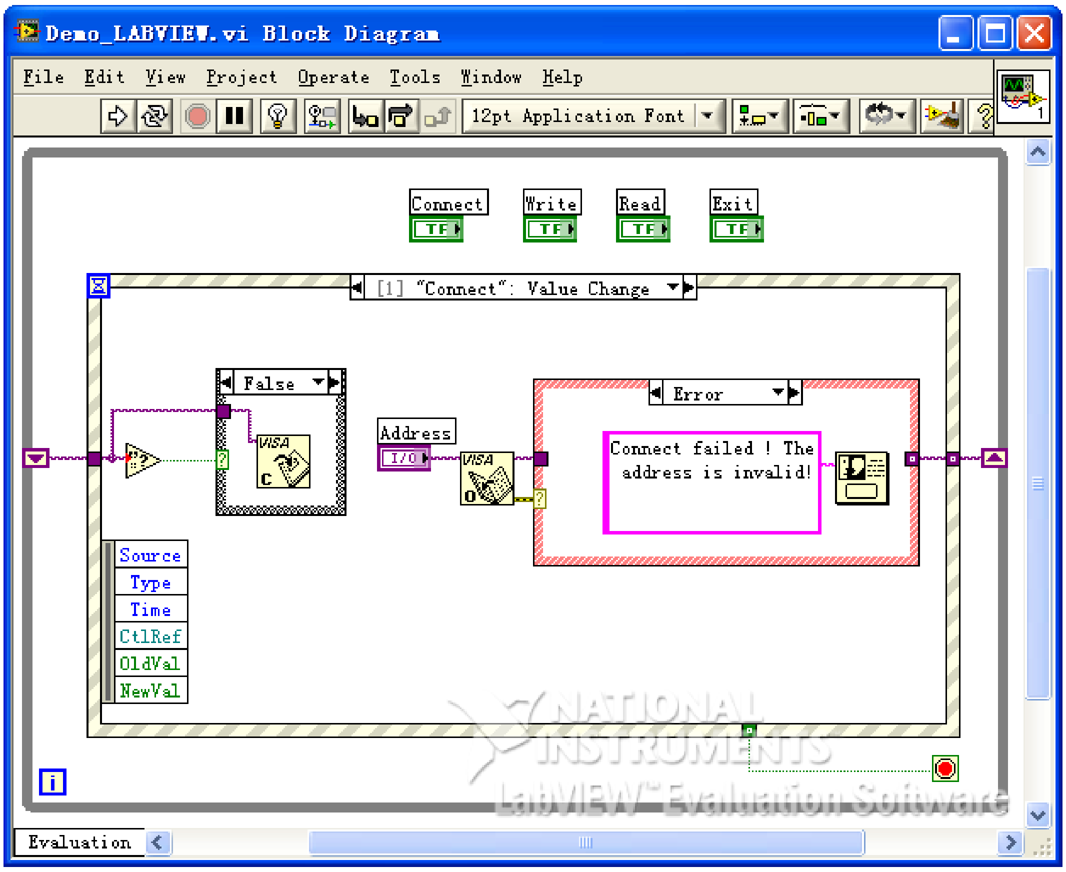
3. Click **Show Block Diagram** in the **Window** menu to create event structure.

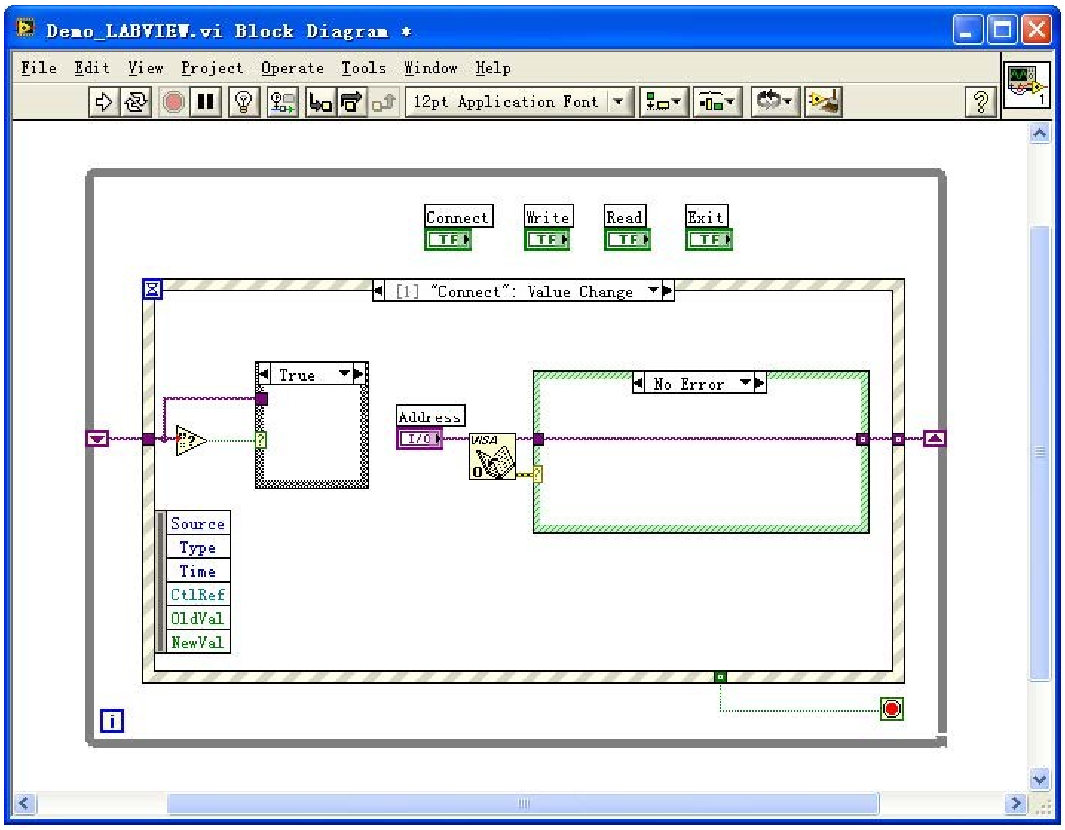

Demo_LABVIEW.vi Block Diagram
File Edit View Project Operate Tools Window Help
12pt Application Font
Address
Command
Return
Connect
Write
Read
Exit
[0] Timeout
Type
Time
NATIONAL INSTRUMENTS LabVIEW Evaluation Software
Evaluation

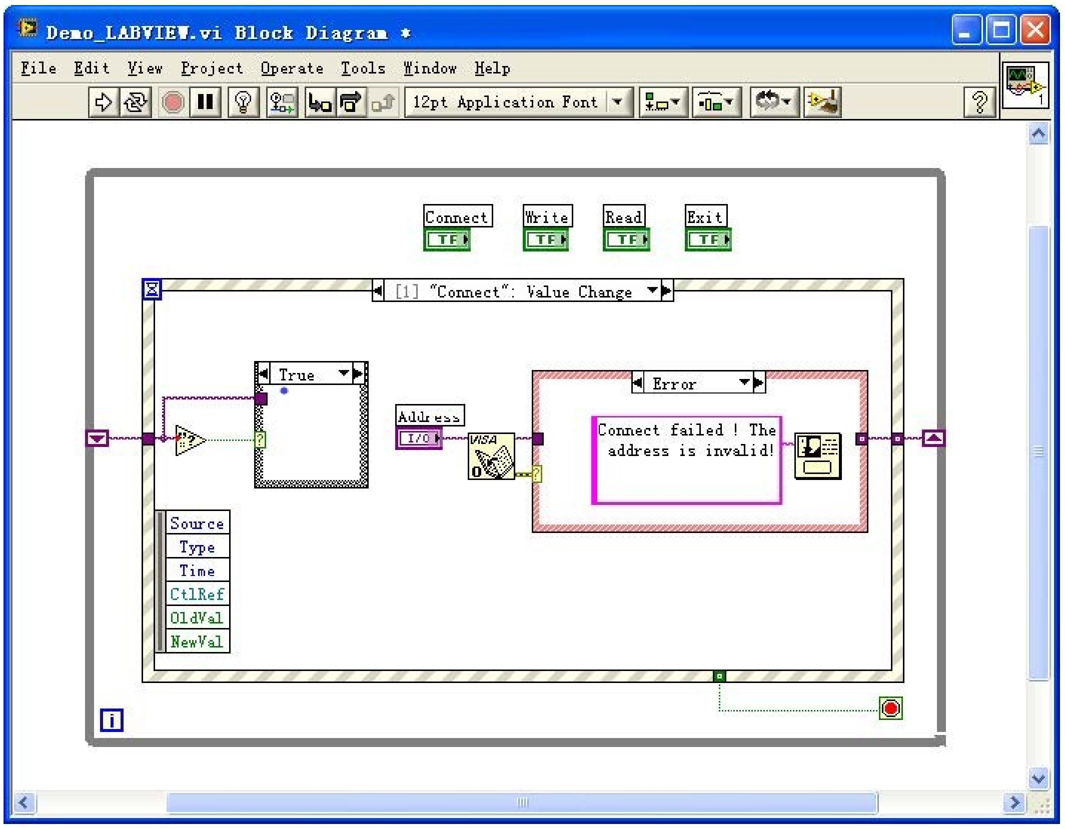
4. Add events, including connecting instrument, write operation, read operation, and exit.

a. Connect the instrument (including error processing):

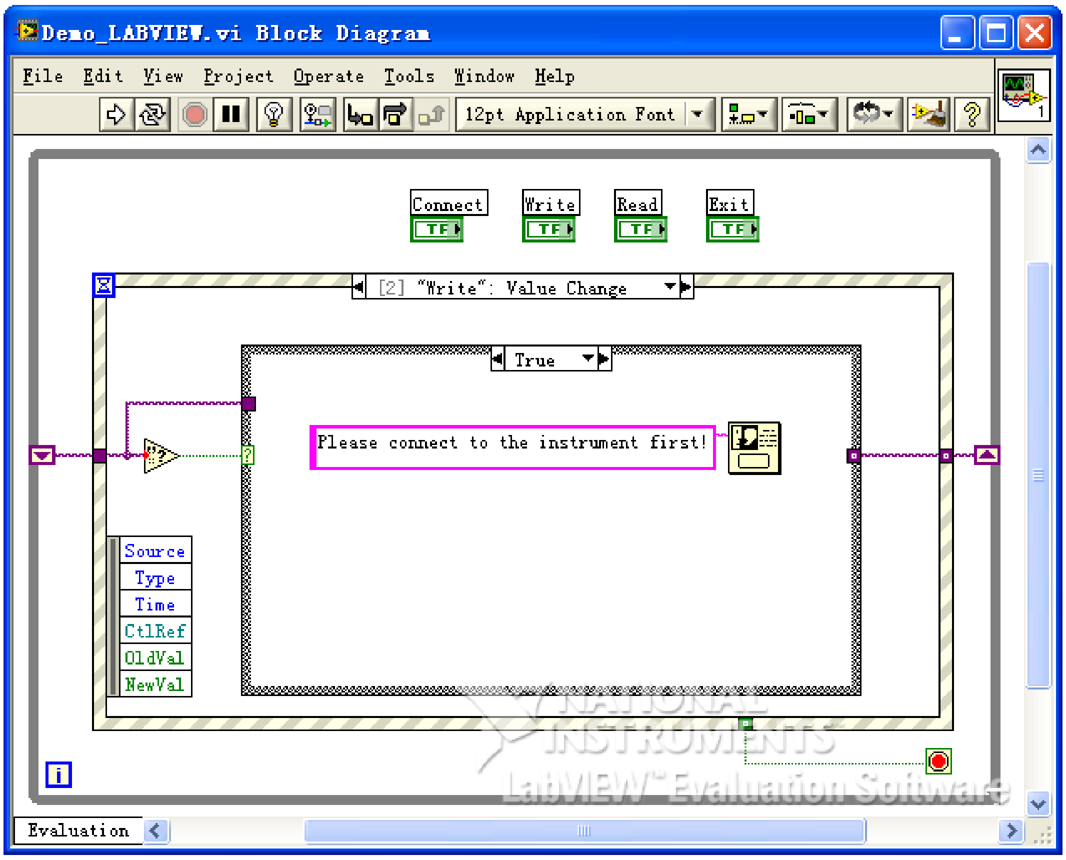

Demo_LABVIEW.vi Block Diagram *
File Edit View Project Operate Tools Window Help
12pt Application Font
Connect
Write
Read
Exit
[1] "Connect": Value Change
False
Address
I/O
VISA
No Error
Source
Type
Time
CtlRef
OldVal
NewVal

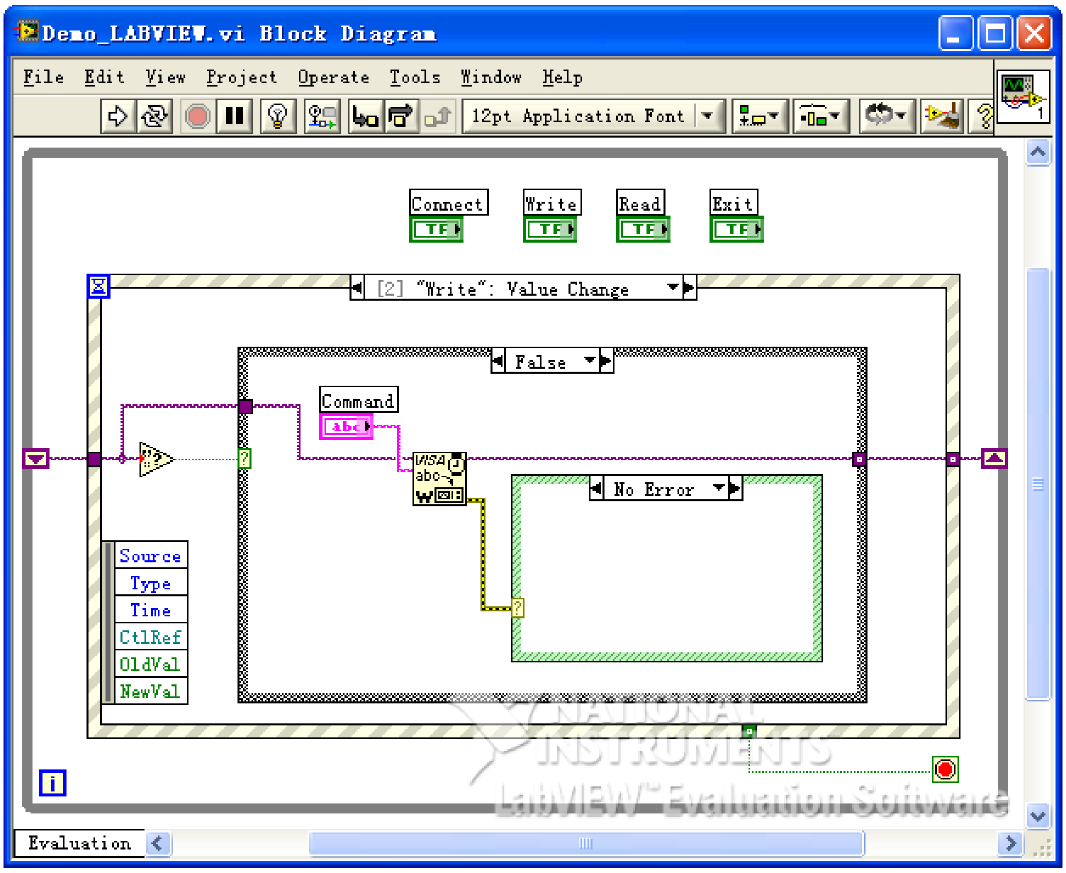

Demo_LABVIEW.vi Block Diagram
File Edit View Project Operate Tools Window Help
12pt Application Font
Connect
Write
Read
Exit
[1] "Connect": Value Change
False
Address
I/O
VISA
Error
Connect failed ! The address is invalid!
Source
Type
Time
CtlRef
OldVal
NewVal
Evaluation


Demo_LABVIEW.vi Block Diagram *
File Edit View Project Operate Tools Window Help
12pt Application Font
Connect
Write
Read
Exit
[1] "Connect": Value Change
True
No Error
Address
I/O
VISA
Source
Type
Time
CtlRef
OldVal
NewVal

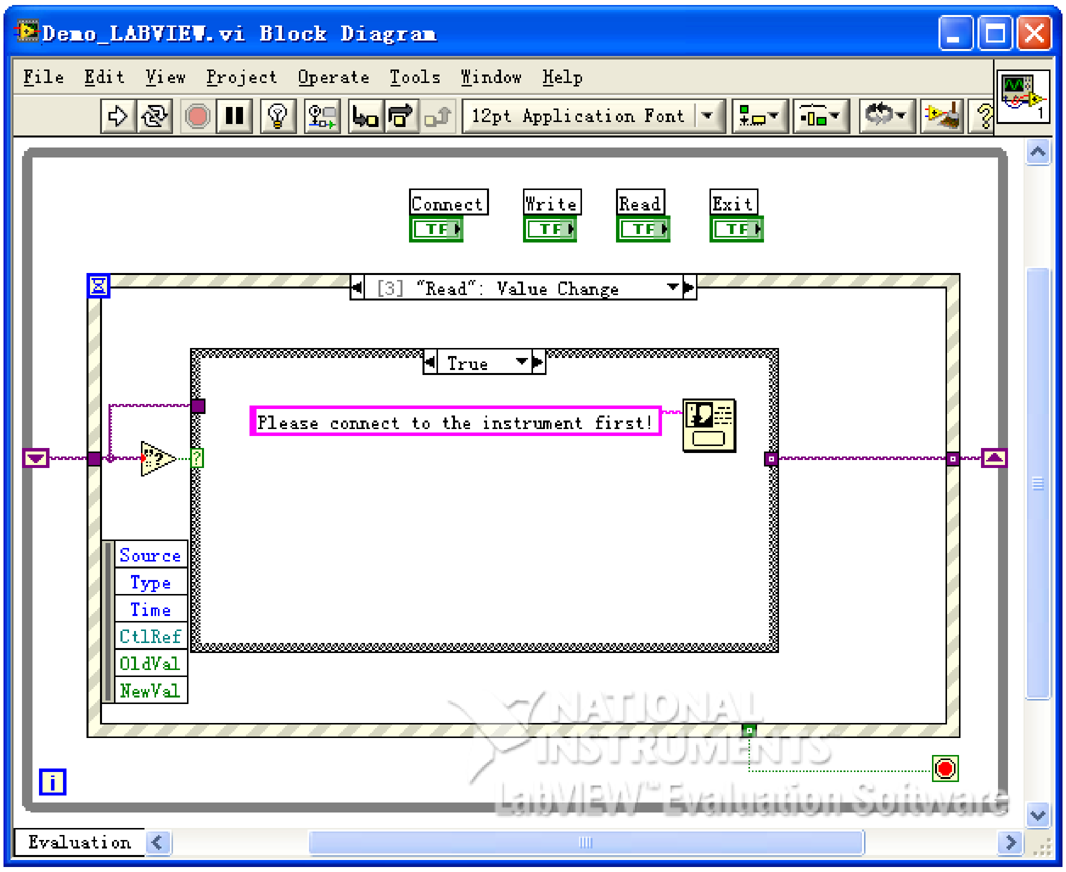

Demo_LABVIEW.vi Block Diagram *
File Edit View Project Operate Tools Window Help
12pt Application Font
Connect
Write
Read
Exit
[1] "Connect": Value Change
True
Error
Connect failed ! The address is invalid!
Address
I/O
VISA
Source
Type
Time
CtlRef
OldVal
NewVal

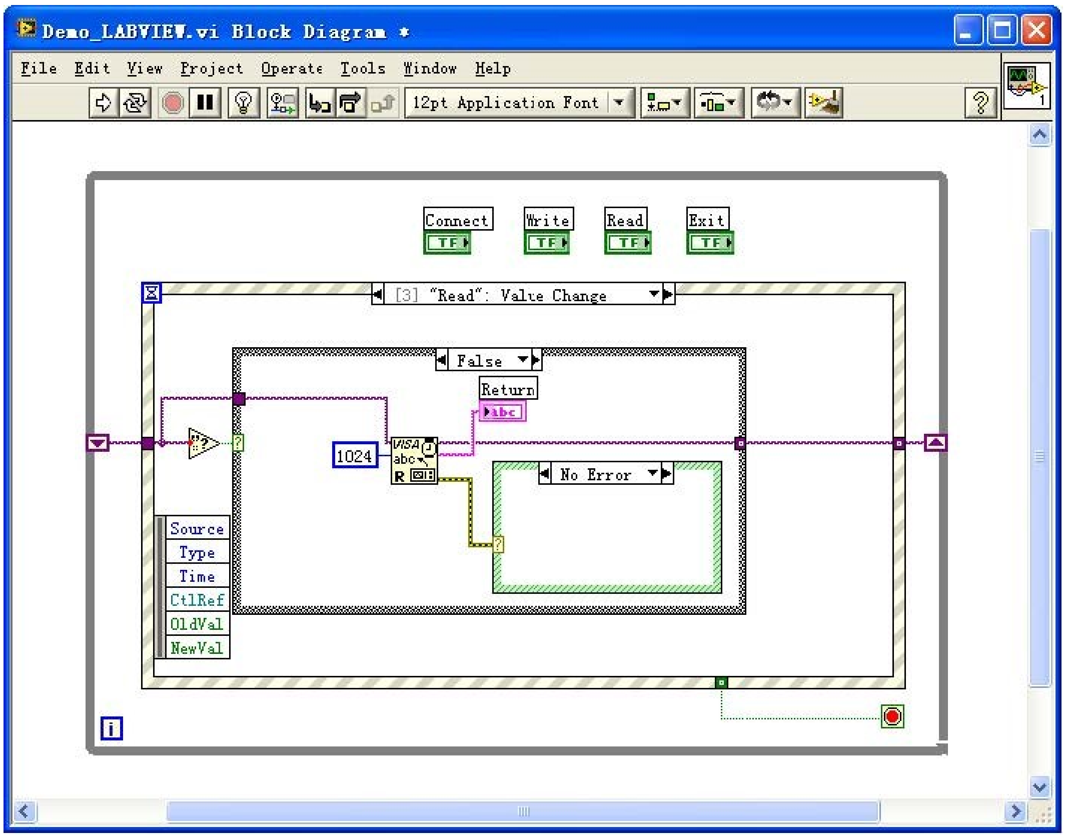
b. Write operation (including error judgment):


Demo_LABVIEW.vi Block Diagram
File Edit View Project Operate Tools Window Help
12pt Application Font
Connect
Write
Read
Exit
[2] "Write": Value Change
True
Please connect to the instrument first!
Source
Type
Time
CtlRef
OldVal
NewVal
Evaluation

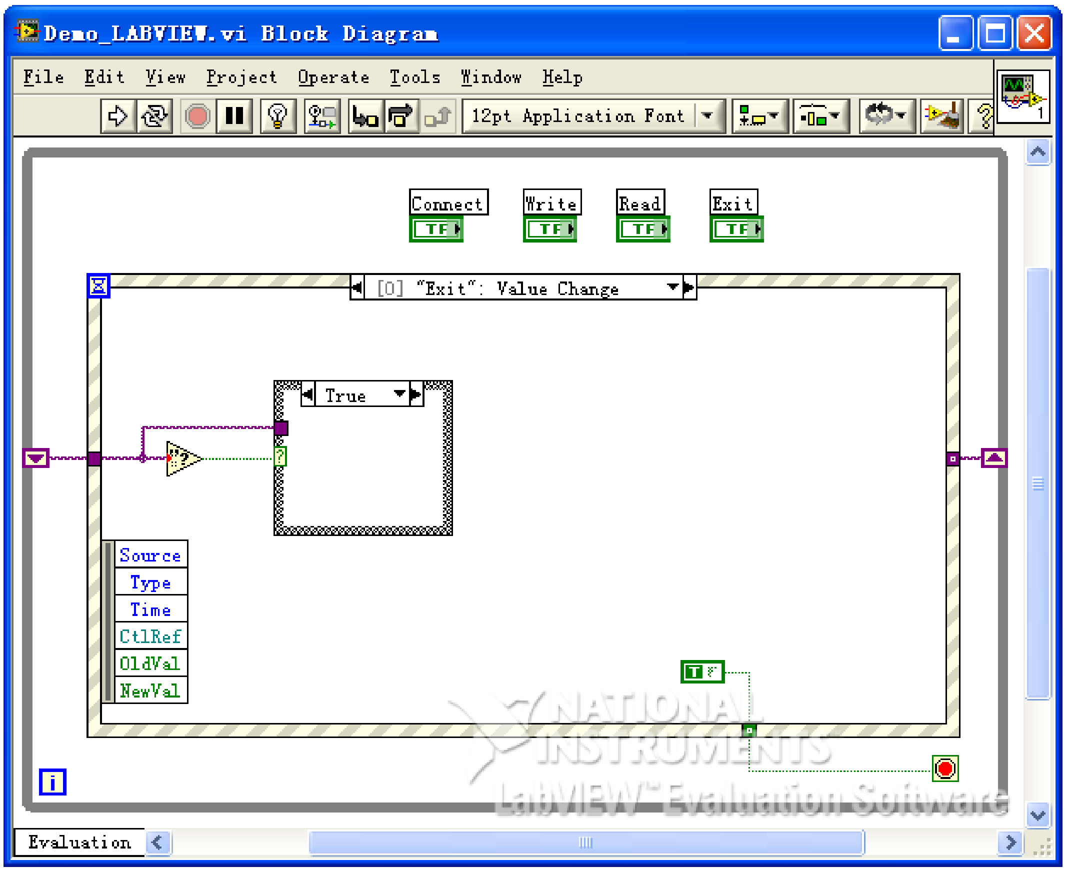


c. Read operation (including error processing):

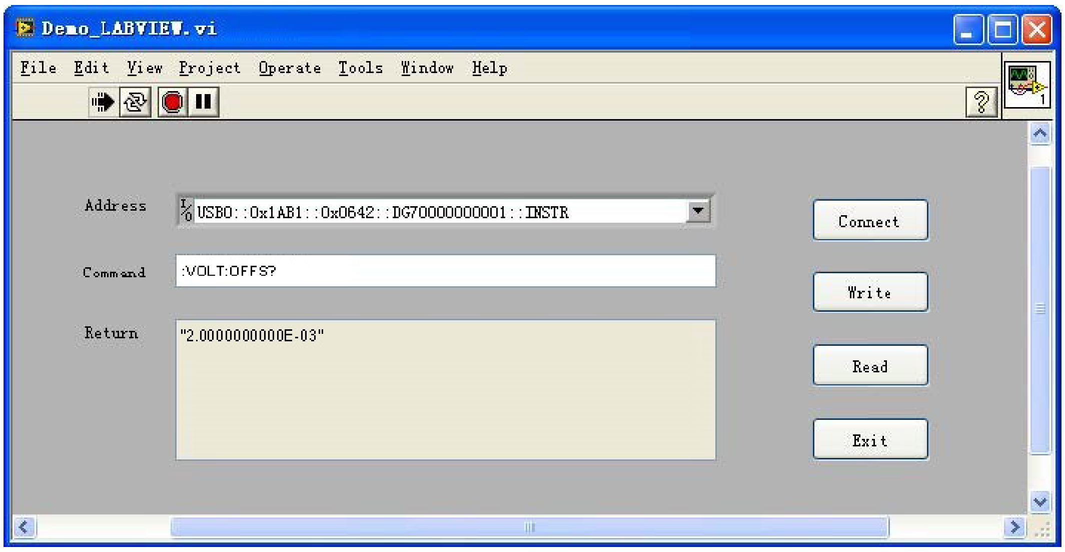

Demo_LABVIEW.vi Block Diagram
File Edit View Project Operate Tools Window Help
12pt Application Font
Connect
Write
Read
Exit
[3] "Read": Value Change
True
Please connect to the instrument first!
Source
Type
Time
CtlRef
OldVal
NewVal
Evaluation


Demo_LABVIEW.vi Block Diagram *
File Edit View Project Operate Tools Window Help
12pt Application Font
Connect
Write
Read
Exit
[3] "Read": Value Change
False
Return
1024
VISA
No Error
Source
Type
Time
CtlRef
OldVal
NewVal

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.



Address USB0::0x1AB1::0x0642::DG70000000001::INSTR

Command :VOLT:OFFS?

Return "2.0000000000E-03"

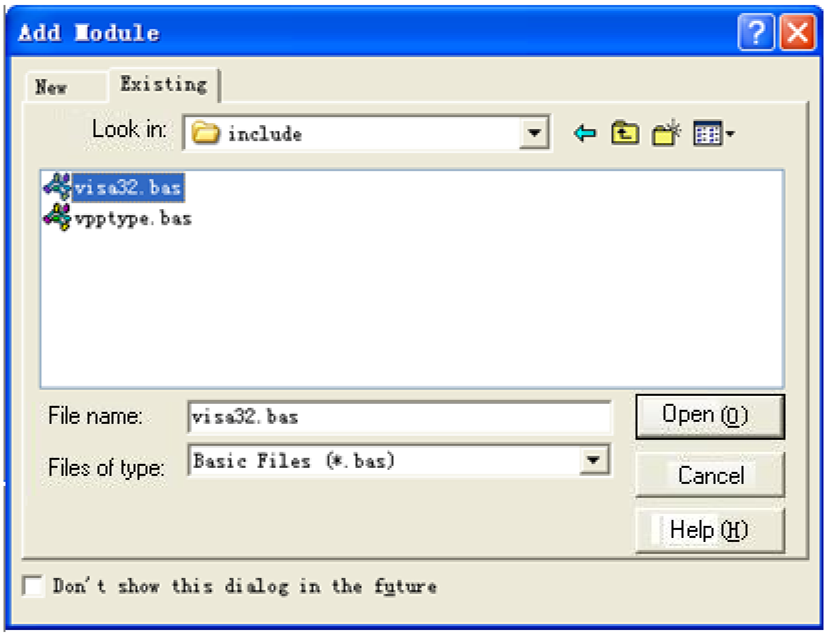
5.5 Visual Basic Programming Example

Program used in this example: Visual Basic 6.0

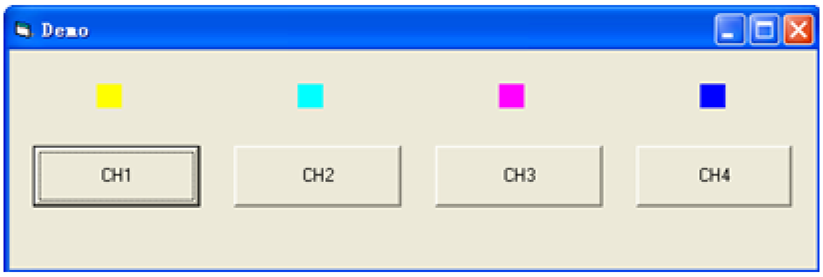
Function realized in this example: control the on/off state of any channel.

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

1. Build a standard executable 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. In the Demo dialog box, add four buttons to represent CH1 to CH4 respectively. Add four Labels (Label1(0), Label1(1),Label1(2),Label1(3)) to represent the status of CH1 to CH4 respectively (when the channel is enabled, it displays the color of the channel; when the channel is disabled, it displays gray), 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 list under Startup Object.

5. Double-click CH1 to enter the programming environment. Add the following codes to control CH1-CH4. 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, ":OUTP1?" + Chr$(10), 0)
' Obtain the status of CH1
Call viVScanf(vi, "%t", strRes)
s32Disp = CInt(strRes)
If (s32Disp = 1) Then
' Send the setting command
Call viVPrintf(vi, "OUTP1 0" + Chr$(10), 0)
Label1(0).ForeColor = &H808080 'Gray
Else
Call viVPrintf(vi, "OUTP1 1" + Chr$(10), 0)
Label1(0).ForeColor = &HFFFF& 'Yellow
End If
' Close the resource
Call viClose(vi)
Call viClose(defrm)
```

6. Save and run the project to obtain a single exe program for demo. When the instrument is correctly connected to the PC, you can control the on/off status of any channel.

5.6 Visual C++ Programming Example

Program used in this example: Visual C++6.0

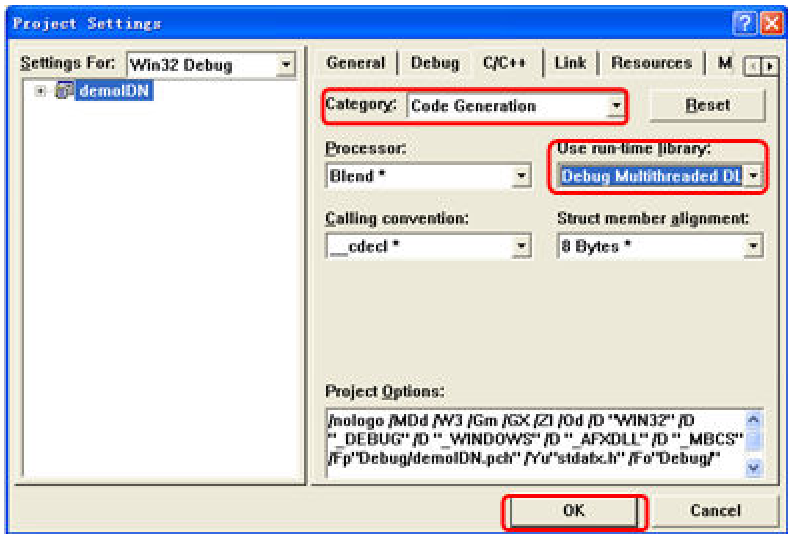
Function realized in this example:search for the instrument address, connect to the instrument, send commands, and read return values.

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

1. Create a MFC project based on a dialog box.

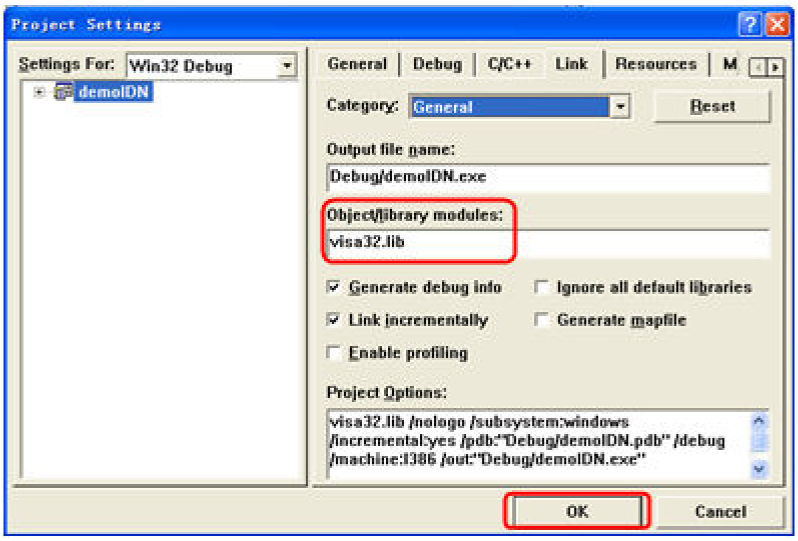
2. Click **Project** > **Settings** to open the **Project Setting** dialog box. In the dialog box, click the **C/C++** tab, select **Code Generation** from the drop-down list under

Category. Choose **Debug Multithreaded DLL** from the drop-down list under **Use run-time library**. Click **OK** to close the dialog box.



Settings For: Win32 Debug
demoIDN
Category: Code Generation
Processor: Blend *
Use run-time library: Debug Multithreaded DL
Calling convention: __cdecl *
Struct member alignment: 8 Bytes *
Project Options:
/nologo /MDd /W3 /Gm /GX /ZI /Od /D "WIN32" /D "_DEBUG" /D "_WINDOWS" /D "_AFXDLL" /D "_MBCS" /Fp"Debug/demoIDN.pch" /Yu"stdafx.h" /Fo"Debug/"

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.



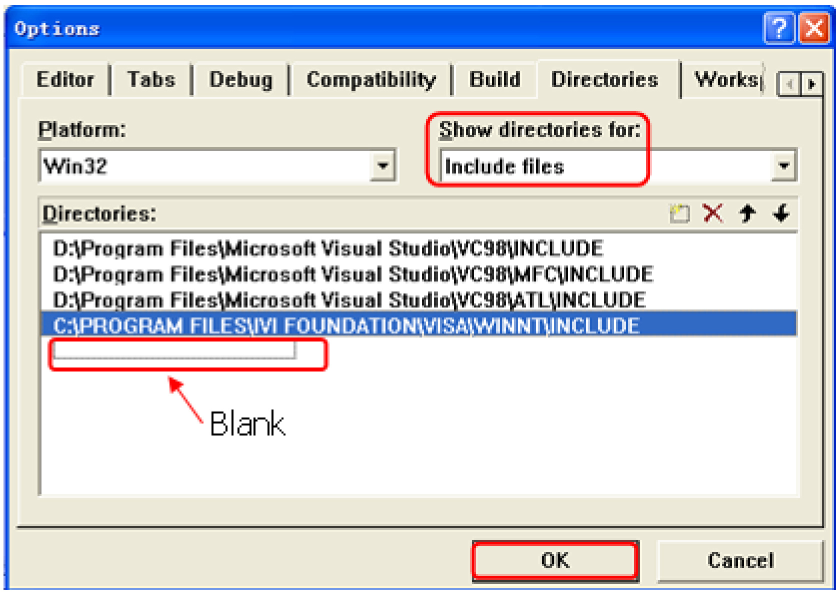
Settings For: Win32 Debug
demoIDN
Category: General
Output file name: Debug/demoIDN.exe
Object/library modules: visa32.lib
✓ Generate debug info
Ignore all default libraries
✓ Link incrementally
Generate mapfile
Enable profiling
Project Options:
visa32.lib /nologo /subsystem:windows /incremental:yes /pdb:"Debug/demoIDN.pdb" /debug /machine:I386 /out:"Debug/demoIDN.exe"

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:

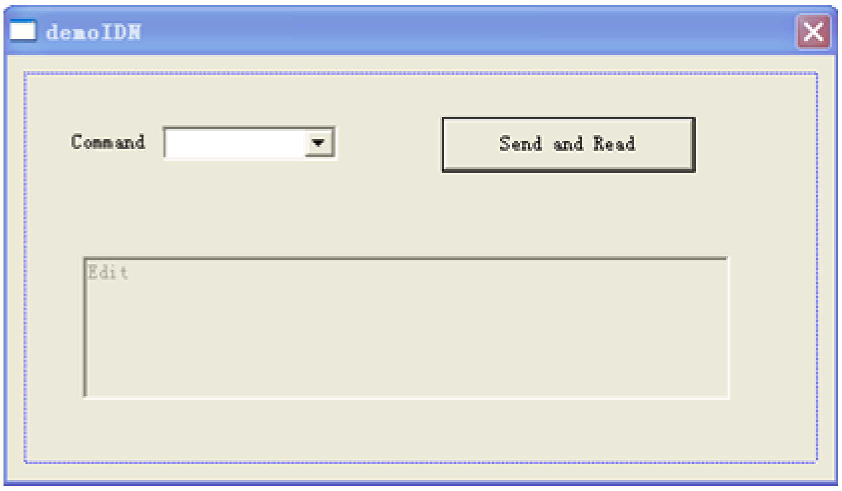
C:\Program Files\IVI Foundation\VISA\WinNT\lib\msc. Click **OK** to close the dialog box.



D:\Program Files\Microsoft Visual Studio\VC98\INCLUDE
D:\Program Files\Microsoft Visual Studio\VC98\MFC\INCLUDE
D:\Program Files\Microsoft Visual Studio\VC98\ATL\INCLUDE
C:\PROGRAM FILES\IVI FOUNDATION\VISA\WINNT\INCLUDE

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.

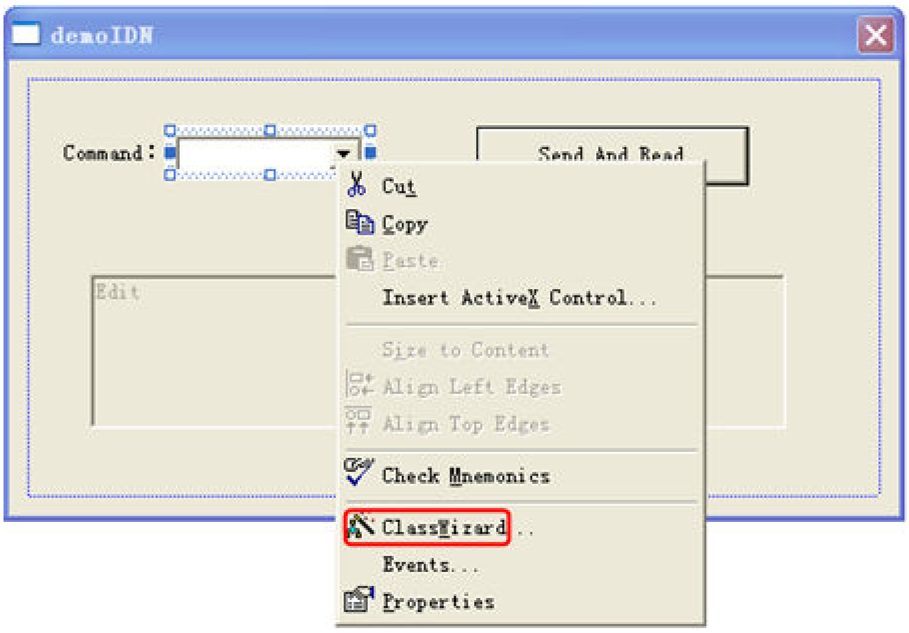
a. Name **Text** as "Command".

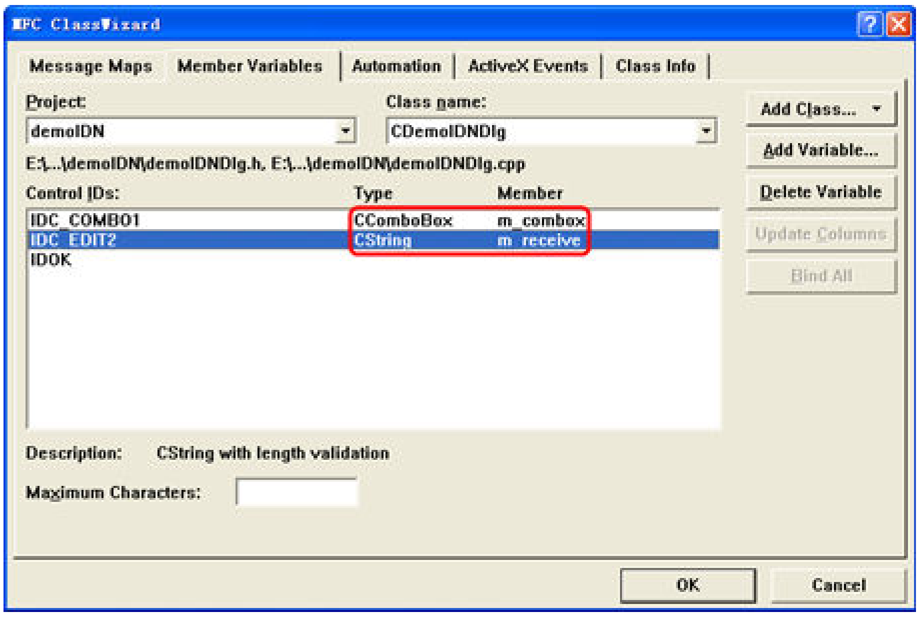
b. Open the **Data** item in the **Combo Box** attribute and input the following command *IDN? manually.

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.





Project: demoIDN

Class name: CDemoIDNDlg

E:\...\demoIDN\demoIDNDlg.h, E:\...\demoIDN\demoIDNDlg.cpp

Control IDs:	Type	Member
IDC_COMBO1	CComboBox	m_combox
IDC_EDIT2	CString	m_receive
IDOK		

Description: CString with length validation

Maximum Characters:

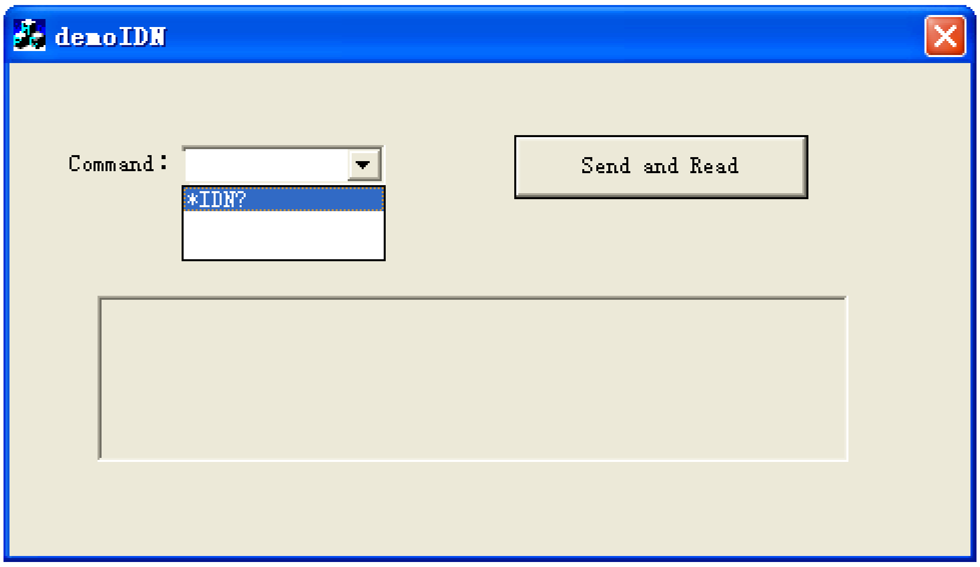
8. Add codes.

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