

UNI-T

Program Manual

UTG Series Programmable Signal Source

June, 2017

UNI-T TECHNOLOGIES , INC.

Warranty and Statement

Copyright

2017 Uni-Trend Technology (China) Co., Ltd.

Brand information

UNI-T is the registered trademark of Uni-Trend Technology (China) Co., Ltd.

Software version

00.00.01

Software upgrade may have some change and add more function, please subscribe

UNI-T website to get the new version or contact **UNI-T**

Statement

- UNI-T products are protected by patents (including obtained and pending) in China and other countries and regions.
- UNI-T reserves the right to change specifications and prices.
- The information provided in this manual supersedes all previous publications.
- Information provided in this manual is subject to change without prior notice.
- UNI-T shall not be liable for any errors that may be contained in this manual. For any incidental or consequential damages arising out of the use or the information and deductive functions provided in this manual.
- No part of this manual shall be photocopied, reproduced or adapted without the prior written permission of UNI-T.

Product certification

UNI-T has certified that the product conforms to China national product standard and industry product standard as well as ISO9001:2008 standard and ISO14001:2004 standard, and further certified that the product complies with relevant standards of other Member of the International Standards Organization.

Contact us

If you have any question or problem, you can contact us

Website : <https://www.uni-trend.com/>

SCPI

SCPI was defined as an additional layer on top of the IEEE 488.2-1987 specification "Standard Codes, Formats, Protocols, and Common Commands". The standard specifies a common syntax, command structure, and data formats, to be used with all instruments. It introduced generic commands (such as CONFigure and MEASure) that could be used with any instrument. These commands are grouped into subsystems. SCPI also defines several classes of instruments. For example, any controllable power supply would implement the same DCPSUPPLY base functionality class. Instrument classes specify which subsystems they implement, as well as any instrument-specific features.

The physical hardware communications link is not defined by SCPI. While it was originally created for the IEEE-488.1 (GPIB) bus, SCPI can also be used with RS-232, RS-422, Ethernet, USB, VXIbus, HiSLIP, etc.

SCPI commands are ASCII textual strings, which are sent to the instrument over the physical layer (e.g., IEEE-488.1). Commands are a series of one or more keywords, many of which take parameters. In the specification, keywords are written CONFigure: The entire keyword can be used, or it can be abbreviated to just the uppercase portion. Responses to query commands are typically ASCII strings. However, for bulk data, binary formats can be used.

This section introduces the format, symbols, parameters, and abbreviations of the SCPI command.

Instruction format

The SCPI command is a tree-like hierarchy consisting of multiple subsystems, each consisting of a root keyword and one or more hierarchical key words. The command line usually begins with a colon ":"; Keywords are separated by the colon ":", followed by optional parameter settings. The command keyword is separated by spaces from the first parameter. The command string must end with a newline <NL> character. Add the question mark "?" after the command line. It is usually indicated that this feature is being queried.

Symbol description

The following four symbols are not part of the SCPI command. It cannot sent with the command, but it is commonly used to supplementary specification.

- **Brace { }** usually contain multiple optional parameter, one of which must be selected when send a command.

Such as DISPlay:GRID:MODE { FULL | GRID | CROSS | NONE} command

- **Vertical bar |** used to separated multiple parameter, one of which must be selected when send a command.

Such as DISPlay:GRID:MODE { FULL | GRID | CROSS | NONE} command

- **Square brackets []** The contents in square brackets (command keywords) can omissible. If the parameter is omitted, the instrument will set the parameter to the default value.

Such as MEASure:NDUTy? [<source>] command presents current channel

- **Triangular braces < >** The parameter in the braces must be replaced with a valid value.

Such as use DISPlay:GRID:BRIGhtness 30 form to send DISPlay:GRID:BRIGhtness <count> command

Parameter description

The parameter in this manual can be divided into five types: Boolean, Integer, Real, Discrete, ASCII string

- **Boolean**

Parameter value can set “ON”(1) or “OFF”(0)

Such as SYSTem:LOCK {{1 | ON} | {0 | OFF}}

- **Integer**

Parameter can take any valid integer value unless there have some other description.

Notes: Do not take decimal point format, otherwise error may occur.

Such as DISPlay:GRID:BRIGhtness <count> command can take integer form 0-100

- **Real**

Parameter can take any valid integer value unless there have some other description.

Such as for command CH1, CHANnel: OFFSet <offset> parameter take integer value.

- **Discrete**

Parameter can only take a specified number or characters.

Such as command DISPlay:GRID:MODE { FULL | GRID | CROSS | NONE} parameter can only take FULL, GRID, CROSS, NONE

- **ASCII string**

String parameter contain all ASCII string sets. Strings must begin and end with paired quotes; it can use single or double quotation marks. The quotation and delimiter can also be part of a string by typing it twice and not adding any characters.

Such as set IP SYST:COMM:LAN:IPAD "192.168.1.10"

Shorthand rule all command can identify capital and small letter, if command need enter shorthand, it should all be capital letter.

Data return is divided into single data and batch data. The single data return is the corresponding parameter type, in which the real return type is presents by the scientific notation method. The part before e retains three figure behind the decimal point, and the e part retains three figure; the batch return must be obey IEEE

488.2# string data format, ‘#’+ the length of character bits[fixed to one character]+ ASCII valid value+ valid data+ end string[‘\n’]

Such as #3123xxxxxxxxxxxxxxxxxxxx\n presents 123 strings batch data return format, ‘3’ presents “123” occupies three character bits.

SCPI explanation

IEEE488.2 General Commands

*IDN?

➤ **Command format:**

*IDN?

➤ **Function description:**

for query manufacture name, power model, product serial number and software version.

➤ **Return format:**

manufacture name, power model, product serial number, software version separated by dot mark.

Notes: The returned model number should be consistent with the nameplate information.

➤ **For example:**

UNI-T Technologies, UTG900, 000000001, 00.00.01

*RST

➤ **Command format:**

*RST

➤ **Function description:**

restore factory settings and clear all the error message, send and receive queue buffers

SYSTem Command

It is used for the basic operation of signal source, including the full QWERTY lock and system data setting.

:SYSTem:LOCK

➤ Command format:

:SYSTem:LOCK {{1 | ON} | {0 | OFF}}

:SYSTem:LOCK?

➤ Function description:

for the full QWERTY lock/unlock

➤ Return format:

query the full QWERTY lock status, 0 presets unlocked, 1 presets locked.

➤ For example:

:SYSTem:LOCK ON	lock the full QWERTY
:SYSTem:LOCK OFF	unlock the full QWERTY
:SYSTem:LOCK?	the query returns to 1 presents locked

:SYSTem:CONFigure

➤ Command format:

:SYSTem:CONFigure <file>

:SYSTem:CONFigure?

➤ Function description:

For read and write configuration files, send the instruction firstly and then send configuration file data to the signal source.

<file> presents configuration file

➤ Return format:

query the current configuration file of the signal source return data

➤ For example:

:SYSTem:CONFigure	writes configuration file data to the signal source and make it load
:SYSTem:CONFigure?	query the current configuration file binary stream of the source return data

:SYSTem:PHASe:MODE

➤ Command format:

:SYSTem:PHASe:MODE {INDependent | SYNChronization}

:SYSTem:PHASe:MODE?

➤ Function description:

Control the phase mode between channels. If synchronous, it presents that the initial phases of the two channels are synchronized; otherwise, the phase are independent.

➤ Return format:

the query returns the phase mode between channels

➤ For example:

:SYSTem:PHASe:MODE INDePendent	set independent phase mode between channels
:SYSTem:PHASe:MODE?	the query returns INDePendent

:SYSTem:LANGuage

➤ **Command format:**

:SYSTem:LANGuage {ENGLish|CHINese}

:SYSTem:LANGuage?

➤ **Function description:**

control the system display language

➤ **Return format:**

query return the system display language

➤ **For example:**

:SYSTem:LANGuage ENGLish

Set English as the system display language

:SYSTem:LANGuage?

the query returns ENGLish

:SYSTem:BEEP

➤ **Command format:**

:SYSTem:BEEP {{1 | ON} | {0 | OFF}}

:SYSTem:BEEP?

➤ **Function description:**

control buzzer switch

➤ **Return format:**

the query returns the buzzer on/off status

➤ **For example:**

:SYSTem:BEEP ON

turn on buzzer

:SYSTem:BEEP?

the query returns 1

:SYSTem:NUMBer:FORMat

➤ **Command format:**

SYSTem:NUMBer:FORMat {COMMa|SPACe|NONE}

:SYSTem:NUMBer:FORMat?

➤ **Function description:**

the delimiter that controls the digital format of the system

➤ **Return format:**

the query returns the delimiter of the digital format of the system

➤ **For example:**

:SYSTem:NUMBer:FORMat NONE

Set no system number format

:SYSTem:NUMBer:FORMat?

the query returns NONE

:SYSTem:BRIGhtness➤ **Command format:**

:SYSTem:BRIGhtness { 10|30|50|70|90|100}

:SYSTem:BRIGhtness?

➤ **Function description:**

control backlight brightness level

➤ **Return format:**

the query returns control backlight brightness level

➤ **For example:**

:SYSTem:BRIGhtness 30

Set backlight brightness to

30%;SYSTem:BRIGhtness?

the query returns 30

:SYSTem:SLEEP:TIME➤ **Command format:**

:SYSTem:SLEEP:TIME { CLOSe | 1 |5 | 15 |30 |60}

:SYSTem:SLEEP:TIME?

➤ **Function description:**

set sleep mode, interval time 5 minute

➤ **Return format:**

the query returns sleep mode

➤ **For example:**

:SYSTem:SLEEP:TIME 1

set the system to automatically sleep mode after 1 minute

:SYSTem:SLEEP:TIME?

the query returns 1

:SYSTem:CYMometer➤ **Command format:**

:SYSTem:CYMometer {{1 | ON} | {0 | OFF}}

:SYSTem:CYMometer?

➤ **Function description:**

Control the system frequency meter switching status

Notes: enable frequency meter function will turn off the channel output synchronous function

➤ **Return format:**

The query returns the system frequency meter switching status, 0 presents off, 1 presents on.

➤ **For example:**

:SYSTem:CYMometer ON

turn on frequency meter

:SYSTem:CYMometer?

the query returns 1

:SYSTem:CYMometer:FREQuency

- **Command format:**
:SYSTem:CYMometer:FREQuency?
- **Function description:**
get the current measured frequency
- **Return format:**
the query returns the current measured frequency of the frequency meter, unit Hz, use scientific notation method returns the data.
- **For example:**
:SYSTem:CYMometer:FREQuency? the query returns 2e+3

:SYSTem:CYMometer:PERiod

- **Command format:**
:SYSTem:CYMometer:PERiod?
- **Function description:**
get the current measurement period of the frequency meter.
- **Return format:**
the query returns the current measured period of the frequency meter, unit S, use scientific notation method returns the data.
- **For example:**
:SYSTem:CYMometer:PERiod? the query returns 2e-3

:SYSTem:CYMometer:DUTY

- **Command format:**
:SYSTem:CYMometer:DUTY?
- **Function description:**
get the current measured duty ratio of the frequency meter.
- **Return format:**
the query returns the current measured duty ratio of the frequency meter, unit is %
- **For example:**
:SYSTem:CYMometer:DUTY? the query returns 20, it presents duty ratio 20%

CHANnel Command

It used to set the relevant channel function of the signal source

:CHANnel<n>:OUTPut

➤ **Command format:**

:CHANnel<n>:OUTPut {{1 | ON} | {0 | OFF}}

:CHANnel<n>:OUTPut?

➤ **Function description:**

set the switch of the channel output

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the status of the channel output, 0 presents off, 1 presents on.

➤ **For example:**

:CHANnel1:OUTPut ON

set to turn on output of channel 1

:CHANnel1:OUTPut?

the query returns 1

:CHANnel<n>:INVersion

➤ **Command format:**

:CHANnel<n>:INVersion {{1 | ON} | {0 | OFF}}

:CHANnel<n>:INVersion?

➤ **Function description:**

set to turn on/off the reverse of the specified channel signal

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the reverse of the specified channel signal, 0 presents off, 1 presents on.

➤ **For example:**

:CHANnel1:INVersion ON

set to turn on the reverse of channel 1

:CHANnel1:INVersion?

the query returns 1

:CHANnel<n>:OUTPut:SYNC➤ **Command format:**

:CHANnel<n>:OUTPut:SYNC {{1 | ON} | {0 | OFF}}

:CHANnel<n>:OUTPut:SYNC?

➤ **Function description:**

set the sync output of the channel

Notes: the device has only one output interface to sync and it can only turn on one channel to sync.

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the sync of the specified channel signal, 0 presents off, 1 presents on.

➤ **For example:**

:CHANnel1:OUTPut:SYNC ON set to turn on the sync of channel 1

:CHANnel1:OUTPut:SYNC? the query returns 1

:CHANnel<n>:LIMit:ENABle➤ **Command format:**

:CHANnel<n>:LIMit:ENABle {{1 | ON} | {0 | OFF}}

:CHANnel<n>:LIMit:ENABle?

➤ **Function description:**

set the amplitude limiting switch of the specified channel signal

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the amplitude limiting switch of the specified channel signal

➤ **For example:**

:CHANnel1:LIMit:ENABle ON

set to turn on the amplitude limiting of the specified channel signal

:CHANnel1:LIMit:ENABle?

the query returns 1

:CHANnel<n>:LIMit:LOWer➤ **Command format:**

:CHANnel<n>:LIMit:LOWer {<voltage>}

:CHANnel<n>:LIMit:LOWer?

➤ **Function description:**

set the lower limit of the specified channel signal

<voltage> voltage of the current channel, unit is V

<n>: channel number, n take 1,2

➤ **Return format:**the query returns the lower limit of the specified channel signal, use scientific notation
method returns the data.➤ **For example:**

:CHANnel1:LIMit:LOWer 2

set the lower limit to 2V of channel 1

:CHANnel1:LIMit:LOWer?

the query returns 2e+0

:CHANnel<n>:LIMit:UPPer➤ **Command format:**

:CHANnel<n>:LIMit:UPPer {<voltage>}

:CHANnel<n>:LIMit:UPPer?

➤ **Function description:**

set the upper limit of the specified channel signal

<voltage> voltage of the current channel, unit is V

<n>: channel number, n take 1,2

➤ **Return format:**the query returns the upper limit of the specified channel signal, use scientific notation
method returns the data.➤ **For example:**

:CHANnel1:LIMit:UPPer 2

set the upper limit to 2V of channel 1

:CHANnel1:LIMit:UPPer?

the query returns 2e+0

:CHANnel<n>:AMPLitude:UNIT➤ **Command format:**

:CHANnel<n>:AMPLitude:UNIT {VPP | VRMS | DBM}

:CHANnel<n>:AMPLitude:UNIT?

➤ **Function description:**

set the amplitude limiting output unit

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the amplitude limiting output unit

➤ **For example:**

:CHANnel1:AMPLitude:UNIT VPP

set the amplitude limiting output unit as VPP of channel 1

:CHANnel1:AMPLitude:UNIT?

the query returns VPP

:CHANnel<n>:LOAD➤ **Command format:**

:CHANnel<n>:LOAD {<resistance>}

:CHANnel<n>:LOAD?

➤ **Function description:**

set the output load of the specified channel signal

<resistance> load resistance value, unit is Ω

<n>: channel number, n take 1,2

Notes: Resistance value range 1~10000, 10000 should be high resistance

➤ **Return format:**

the query returns the load resistance value of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:LOAD 50

set the load resistance value to 50 Ω of channel 1

:CHANnel1:LOAD?

the query returns 50e+0

:CHANnel<n>:BASE:WAVe➤ **Command format:**

:CHANnel<n>:BASE:WAVe { SIne | SQUare | PULSe | RAMP | ARB | NOISe | DC }

:CHANnel<n>:BASE:WAVe?

➤ **Function description:**

Set the fundamental wave type of the specified channel signal. They are sine wave, square wave, pulse wave, triangular wave, random wave, noise and direct current

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the fundamental wave type of the specified channel signal

➤ **For example:**

:CHANnel1:BASE:WAVe SIne set the sine wave of channel 1

:CHANnel1:BASE:BWAVe? the query returns SIne

:CHANnel<n>:BASE:FREQuency➤ **Command format:**

:CHANnel<n>:BASE:FREQuency {<freq>}

:CHANnel<n>:BASE:FREQuency?

➤ **Function description:**

set the output frequency of the specified channel signal

<freq> the frequency value, unit is Hz (1e-6s ~ the maximum allowable frequency of the current waveform)

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the output frequency of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:BASE:FREQuency 2000 set the output frequency to 2KHz of channel 1

:CHANnel1:BASE:FREQuency? the query returns 2e+3

:CHANnel<n>:BASE:PERiod➤ **Command format:**

:CHANnel<n>:BASE:PERiod { <period>}

:CHANnel<n>:BASE:PERiod?

➤ **Function description:**

set the output period of the specified channel signal

<period> output period, unit is S. If it is sine wave, range (the current maximum allowable time is $\sim 1\text{e}3\text{s}$)

<n>: channel number, n take 1,2

➤ **Return format:**the query returns the upper limit of the specified channel signal, use scientific notation
method returns the data.➤ **For example:**

:CHANnel1:BASE:PERiod 0.002

set the output period to 2ms of channel 1

:CHANnel1:BASE:PERiod ?

the query returns $2\text{e-}3$ **:CHANnel<n>:BASE:PHASe**➤ **Command format:**

:CHANnel<n>:BASE:PHASe { <phase>}

:CHANnel<n>:BASE:PHASe?

➤ **Function description:**

set the output phase of the specified channel signal

<phase> unit is $^{\circ}$, range $-360\sim 360$.

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the output phase of the specified channel signal

➤ **For example:**

:CHANnel1:BASE:PHASe 20

set the output phase to 20° of channel 1

:CHANnel1:BASE:PHASe?

the query returns 20

:CHANnel<n>:BASE:AMPLitude➤ **Command format:**

:CHANnel<n>:BASE:AMPLitude { <amp> }

:CHANnel<n>:BASE:AMPLitude?

➤ **Function description:**

set the amplitude limiting output of the specified channel signal

<amp> presents voltage, unit is the current channel

1mVpp ~ the maximum output load of the current channel

If the unit is VPP, the maximum output load value= the current load *20/(50+ the current load)

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the amplitude limiting output of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:BASE:AMPLitude 2	set the amplitude limiting output to 2V of channel 1
:CHANnel1:BASE:AMPLitude?	the query returns 2e+0

:CHANnel<n>:BASE:OFFSet➤ **Command format:**

:CHANnel<n>:BASE:OFFSet { <voltage> }

:CHANnel<n>:BASE:OFFSet?

➤ **Function description:**

set the DC offset output of the specified channel signal

<voltage> unit isV, range: 0~±the maximum on-load DC voltage of the current

the maximum on-load DC voltage of the current = the current on-load *10/(50+ the current on-load) - the minimum on-load AC voltage of the current/2; the

minimum AC voltage is2mVpp, DC voltage mode take;

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the DC offset output of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:BASE:OFFSet 2	set the DC offset output to 2V of channel 1
:CHANnel1:BASE:OFFSet?	the query returns 2e+0

:CHANnel<n>:BASE:HIGH➤ **Command format:**

:CHANnel<n>:BASE:HIGH { <voltage> }

:CHANnel<n>:BASE:HIGH?

➤ **Function description:**

set the high signal output value of the specified channel signal

<voltage> unit is the current channel

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the high signal output value of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:BASE:HIGH 2

set the high signal output value 2V of channel 1

:CHANnel1:BASE:HIGH?

the query returns 2e+0

:CHANnel<n>:BASE:LOW➤ **Command format:**

:CHANnel<n>:BASE:LOW { <voltage> }

:CHANnel<n>:BASE:LOW?

➤ **Function description:**

set the low signal output value of the specified channel signal

<voltage> unit is the current channel

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the low signal output value of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:BASE:LOW 2

set the low signal output value to 2V of channel 1

:CHANnel1:BASE:LOW?

the query returns 2e+0

:CHANnel<n>:BASE:DUTY➤ **Command format:**

:CHANnel<n>:BASE:DUTY { <duty> }

:CHANnel<n>:BASE:DUTY?

➤ **Function description:**

set the duty ratio of the specified channel signal

<duty> presents duty ratio, unit is %, range 0~100

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the duty ratio of the specified channel signal

➤ **For example:**

:CHANnel1:BASE:DUTY 20

set the low signal output value to 20% of channel 1

:CHANnel1:BASE:DUTY?

the query returns 20

:CHANnel<n>:RAMP:SYMMetry➤ **Command format:**

:CHANnel<n>:RAMP:SYMMetry { < symmetry > }

:CHANnel<n>:RAMP:SYMMetry?

➤ **Function description:**

set ramp signal output symmetry of the specified channel signal

< symmetry > unit is %, range 0~100

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns ramp signal output symmetry of the specified channel signal

➤ **For example:**

:CHANnel1:RAMP:SYMMetry 20

set the ramp signal symmetry value to 20% of channel 1

:CHANnel1:RAMP:SYMMetry?

the query returns 20

:CHANnel<n>:PULSe:RISe➤ **Command format:**

:CHANnel<n>:PULSe:RISe { <width> }

:CHANnel<n>:PULSe:RISe?

➤ **Function description:**

set pulse wave rising edge pulse width of the specified channel signal

<width> presents pulse width, unit is S

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns ramp signal output symmetry of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:PULSe:RISe 0.002

set the rising edge pulse width value to 2ms of channel 1

:CHANnel1:PULSe:RISe?

the query returns 2e-3

:CHANnel<n>:PULSe:FALL➤ **Command format:**

:CHANnel<n>:PULSe:FALL {<width>}

:CHANnel<n>:PULSe:FALL?

➤ **Function description:**

set pulse wave falling edge pulse width of the specified channel signal

<width> presents pulse width, unit is S

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns pulse wave falling edge pulse width of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:PULSe:FALL 0.002

set the falling edge pulse width value to 2ms of channel 1

:CHANnel1:PULSe:FALL?

the query returns 2e-3

:CHANnel<n>:MODE➤ **Command format:**

:CHANnel<n>:MODE {CONTINUE | AM | PM | FM | FSK | Line | Log }

:CHANnel<n>:MODE?

➤ **Function description:**

set the signal mode of the specified channel signal, they are CONTINUE 、AM、PM 、FM、FSK、Line、Log

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the signal mode of the specified channel signal

➤ **For example:**

:CHANnel1:MODE AM

set the amplitude modulation mode of channel 1

:CHANnel1:MODE?

the query returns AM

:CHANnel<n>:MODulate:WAVE➤ **Command format:**

:CHANnel<n>:MODulate:WAVE { SINE|SQUare|UPRamp|DNRamp|ARB|NOISe }

:CHANnel<n>:MODulate:WAVE?

➤ **Function description:**

set modulating wave type of the specified channel signal, they are sine wave, square wave, upper triangle, lower triangle, random wave and noise.

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the modulating wave type of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:MODulate:WAVe SINE

set the amplitude modulation mode to sine wave of channel 1

:CHANnel1:MODulate:WAVe?

the query returns SINE

:CHANnel<n>:MODulate:FREQuency

➤ **Command format:**

:CHANnel<n>:MODulate:FREQuency {<freq>}

:CHANnel<n>:MODulate:FREQuency?

➤ **Function description:**

set modulating frequency type of the specified channel signal, use scientific notation method returns the data.

<freq> presents frequency, unit isHz

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns the frequency wave type of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

:CHANnel1:MODulate:FREQuency 2000

set the frequency wave to 2kHz of channel 1

:CHANnel1:MODulate:FREQuency?

the query returns 2e+3

:CHANnel<n>:MODulate:ARB:INDex

➤ **Command format:**

:CHANnel<n>:MODulate:ARB:INDex {<index>}

:CHANnel<n>:MODulate:ARB:INDex?

➤ **Function description:**

set the random wave serial number saved by the specified load signal source

<index> presents random wave serial number

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns random wave serial number

➤ **For example:**

:CHANnel1:MODulate:ARB:IND 2

Channel 1 loads the second modulated random wave saved by the signal source

:CHANnel1:MODulate:ARB:IND?

the query returns 2

:CHANnel<n>:MODulate:ARB:SOURce

➤ **Command format:**

:CHANnel<n>:MODulate:ARB:SOURce { INTernal|EXTernal }

:CHANnel<n>:MODulate:ARB:SOURce?

➤ **Function description:**

set the specified channel modulation of random wave source, internal and external

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns random wave

➤ **For example:**

:CHANnel1:MODulate:ARB:SOURce INTernal

set the random wave as external of channel 1

:CHANnel1:MODulate:ARB:SOURce?

the query returns INTernal

:CHANnel<n>:MODulate:DEPTh➤ **Command format:**

:CHANnel<n>:MODulate:DEPTh { <depth> }

:CHANnel<n>:MODulate:DEPTh?

➤ **Function description:**

set the modulation depth of the specified channel.

<depth> presents the modulation depth, unit is %, range 0% ~ 100%, AM modulation depth 0% ~ 120%

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns the modulation depth

➤ **For example:**

:CHANnel1:MODulate:DEPTh 50

set the modulation depth to 50% of channel 1

:CHANnel1:MODulate:DEPTh?

the query returns 50

:CHANnel<n>:ARB:INDex➤ **Command format:**

:CHANnel<n>:ARB:INDex { <index > }

:CHANnel<n>:ARB:INDex?

➤ **Function description:**

set the random wave serial number saved by the specified load signal source

<index > presents random wave serial number

<n>: channel number, n take 1,2

➤ **Return format:**

the query returns random wave

➤ **For example:**

:CHANnel1:ARB:IND 2

Channel 1 loads the second modulated random wave saved by the signal source

:CHANnel1:ARB:IND?

the query returns 2

:CHANnel<n>:ARB:SOURce➤ **Command format:**

```
:CHANnel<n>:ARB:SOURce { INTernal|EXTernal }
```

```
:CHANnel<n>:ARB:SOURce?
```

➤ **Function description:**

set the specified channel modulation of random wave source, internal and external

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns random wave

➤ **For example:**

```
:CHANnel1:ARB:SOURce INTernal
```

Channel 1 loads the second modulated random wave saved by the signal source

```
:CHANnel1:ARB:SOURce?
```

the query returns INTernal

:CHANnel<n>:FM:FREQuency:DEV➤ **Command format:**

```
:CHANnel<n>:FM:FREQuency:DEV { <freq> }
```

```
:CHANnel<n>:FM:FREQuency:DEV?
```

➤ **Function description:**

set frequency deviation of the specified channel

<freq> presents frequency deviation, unit is Hz. 0Hz ~ the current fundamental frequency

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns the frequency deviation of the specified channel signal, use scientific notation method returns the data.

➤ **For example:**

```
:CHANnel<n>:FM:FREQuency:DEV 2000
```

set the frequency deviation to 2kHz of the channel 1

```
:CHANnel<n>:FM:FREQuency:DEV?
```

the query returns 2e+3

:CHANnel<n>:PM:PHASe:DEV➤ **Command format:**

:CHANnel<n>:PM:PHASe:DEV { <phase>}

:CHANnel<n>:PM:PHASe:DEV?

➤ **Function description:**

set the phase deviation output of the specified channel

< phase > presents deviation output, unit is°, range 0~360

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns the phase deviation

➤ **For example:**

:CHANnel<n>:PM:PHASe:DEV 30

set the phase deviation to 30° of the specified channel 1

:CHANnel<n>:PM:PHASe:DEV?

the query returns 30

:CHANnel<n>:FSK:HOPPing:FREQuency➤ **Command format:**

:CHANnel<n>:FSK:HOPPing:FREQuency { <freq>}

:CHANnel<n>:FSK:HOPPing:FREQuency?

➤ **Function description:**

set the frequency hopping of the specified channel

< freq > presents frequency, unit isHz

<n>: channel number, n take 1, 2

➤ **Return format:**the query returns the frequency hopping of the specified channel, use scientific notation
method returns the data.➤ **For example:**

:CHANnel<n>:FSK:HOPP:FREQ 2000

set the frequency hopping to 2kHz of channel 1

:CHANnel<n>:FSK:HOPP:FREQ?

the query returns 2e+3

:CHANnel<n>:SWEep:FREQuency:STARt

➤ **Command format:**

:CHANnel<n>:SWEep:FREQuency:STARt { <freq>}

:CHANnel<n>:SWEep:FREQuency:STARt?

➤ **Function description:**

set the sweep frequency starting frequency of the specified channel

< freq > presents frequency , unit is Hz

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns the sweep frequency starting frequency of the specified channel, use scientific notation method returns the data.

➤ **For example:**

:CHANnel<n>:SWE:FREQ:STAR 2000

set the sweep frequency starting frequency to 2kHz of channel 1

:CHANnel<n>:SWE:FREQ:STAR?

the query returns 2e+3

:CHANnel<n>:SWEep:FREQuency:STOP

➤ **Command format:**

:CHANnel<n>:SWEep:FREQuency:STOP { <freq>}

:CHANnel<n>:SWEep:FREQuency:STOP?

➤ **Function description:**

set the sweep frequency cutoff frequency of the specified channel

< freq > presents frequency , unit is Hz

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns the sweep frequency cutoff frequency of the specified channel, use scientific notation method returns the data.

➤ **For example:**

:CHANnel<n>:SWE:FREQ:STOP 2000

set the sweep frequency cutoff frequency to 2kHz of channel 1

:CHANnel<n>:SWE:FREQ:STOP?

the query returns 2e+3

:CHANnel<n>:SWEEP:TIME

➤ **Command format:**

:CHANnel<n>:SWEEP:TIME { <time> }

:CHANnel<n>:SWEEP:TIME?

➤ **Function description:**

set the sweep frequency scanning time of the specified channel

< time > unit is S, range 1ms ~ 500s

<n>: channel number, n take 1, 2

➤ **Return format:**

the query returns the sweep frequency scanning time of the specified channel, use scientific notation method returns the data.

➤ **For example:**

:CHANnel<n>:SWEEP:TIME 2

set the sweep frequency scanning time to 2s of channel 1

:CHANnel<n>:SWEEP:TIME?

the query returns 2e+0

WARB Command

Instructions for writing arbitrary waveform files, including basic arbitrary waveform and modulated arbitrary waveform write configurations.

:WARB<n>:MODulate

➤ **Command format:**

:WARB<n>:MODulate <arb file>

➤ **Function description:**

Used to write to modulate arbitrary waveform, waveform data fixed 4K points, send the instruction firstly, and then send arbitrary waveform file data to the signal source.

<arb file> presents arbitrary waveform file

➤ **For example:**

:WARB1:MODulate

set the modulate arbitrary waveform of channel 1

:WARB<n>:CARRier

➤ **Command format:**

:WARB<n>:CARRier <arb file>

➤ **Function description:**

Used to write to modulate arbitrary waveform, waveform data fixed 4K points, send the instruction firstly, and then send arbitrary waveform file data to the signal source.

<arb file> presents arbitrary waveform file

➤ **For example:**

:WARB1:CARRier

write the modulate arbitrary waveform of channel 1

DISPlay Command

Used for to display image information of the signal source

:DISPlay?

➤ **Command format:**

:DISPlay?

➤ **Function description:**

used to query the image data on the current screen of an oscilloscope

➤ **Return format:**

the query returns image data, and the returned data conforms to IEEE 488.2 # format binary data.

➤ **For example:**

:DISPlay?

the query returns image data

data format: #800012345+ bitmap data

KEY Command

For controlling the keys and knobs on the power operation panel

:KEY:<key>

➤ **Command format:**

:KEY:<key>

:KEY:<key>:LOCK { {1 | ON} | {0 | OFF} }

:KEY:<key>:LOCK?

:KEY:<key>:LED?

➤ **Function description:**

Used to set the lock/unlock function.

<key> defined and described see in Appendix 1: <key> format

➤ **Return format:**

The query returns a key locked statue or a state with LED keystroke lights.

Lock statue: 0 presents unlocked, 1 presents locked

LED statue: 0 presents light off, 1 presents light on (green) , 2 presents light on (red)

➤ **For example:**

:KEY:AUTO

set the oscilloscope control value automatically

:KEY:AUTO:LOCK ON/OFF

locked/unlocked button

:KEY:AUTO:LOCK?

the query returns lock statue, 1 presents locked

:KEY:AUTO:LED?

the query returns LED statue, 0 presents light off

Appendix 1: <key> format

Command keywords	Function description	LED
Wave	Wave type	✓
Mode	Output mode	
Utility	System	
Symbol	Numeric key symbol	
Dot	Number key decimal point	
NUM0	Numeric key 0	
NUM1	Numeric key1	
NUM2	Numeric key2	
NUM3	Numeric key3	
NUM4	Numeric key4	
NUM5	Numeric key5	
NUM6	Numeric key6	
NUM7	Numeric key 7	
NUM8	Numeric key8	
NUM9	Numeric key9	
Up	Up arrow key	
Down	Down arrow key	
Left	Left arrow key	
Right	Right arrow key	
OK	Confirm key	
CH1	Channel 1 key	✓
CH2	Channel 2 key	✓
F1	Select the first item of the current menu	
F2	Select the second item of the current menu	
F3	Select the third item of the current menu	
F4	Select the fourth item of the current menu	
F5	Select the fifth item of the current menu	
F6	Select the sixth item of the current menu	

Appendix 2: key lock statue

Tag number	Key	Status
0	Wave	0 presents unlocked, 1 presents locked
1	Mode	0 presents unlocked, 1 presents locked
2	Utility	0 presents unlocked, 1 presents locked
3	Symbol	0 presents unlocked, 1 presents locked
4	Dot	0 presents unlocked, 1 presents locked
5	NUM0	0 presents unlocked, 1 presents locked
6	NUM1	0 presents unlocked, 1 presents locked
7	NUM2	0 presents unlocked, 1 presents locked
8	NUM3	0 presents unlocked, 1 presents locked
9	NUM4	0 presents unlocked, 1 presents locked
10	NUM5	0 presents unlocked, 1 presents locked
11	NUM6	0 presents unlocked, 1 presents locked
12	NUM7	0 presents unlocked, 1 presents locked
13	NUM8	0 presents unlocked, 1 presents locked
14	NUM9	0 presents unlocked, 1 presents locked
15	Up	0 presents unlocked, 1 presents locked
16	Down	0 presents unlocked, 1 presents locked
17	Left	0 presents unlocked, 1 presents locked
18	Right	0 presents unlocked, 1 presents locked
19	OK	0 presents unlocked, 1 presents locked
20	CH1	0 presents unlocked, 1 presents locked
21	CH2	0 presents unlocked, 1 presents locked
22	F1	0 presents unlocked, 1 presents locked
23	F2	0 presents unlocked, 1 presents locked
24	F3	0 presents unlocked, 1 presents locked
25	F4	0 presents unlocked, 1 presents locked
26	F5	0 presents unlocked, 1 presents locked
27	F6	0 presents unlocked, 1 presents locked

