Instruments.uni-trend.com

# MSO1000HD Series High-Resolution Oscilloscopes

## Programming Manual

V1.0

December 2024

# Warranty and Statement

## Copyright

Copyright © 2024 by Uni-Trend Technology (China) Co., Ltd.

## Trademark

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

## Document No.

20241226

## Software Version

1.00.0009

Software upgrade may have function updates and changes, please subscribe **UNI-T** website to get the latest version or contact **UNI-T**.

## Statement

- UNI-T products are protected by patent rights in China and foreign countries, including issued and pending patents.
- UNI-T reserves the rights to any product specification and pricing changes.
- UNI-T reserves all rights. Licensed software products are properties of UNI-Trend and its subsidiaries or suppliers, which are protected by national copyright laws and international treaty provisions. Information in this manual supersedes all previously published versions.
- Without the written permission of **UNI-T**, this manual cannot be photocopied, reproduced or adapted.

## Product Certificate

**UNI-T** has certified that the product conforms to China national product standard and industry product standard as well as ISO9001: 2015standard and ISO14001: 2015standard. UNI-T will go further to certificate products to meet the standard of other members of the international standards organization.

# SCPI

SCPI (Standard Commands for Programmable Instruments) is a standard command set based on the existing standards IEEE 488.1 and IEEE 488.2. And follow the IEEE754 standard floating-point arithmetic rules, ISO646 information exchange 7 bits code symbol (equivalent to ASCII programming) and other standard standardized instrument programming language.

## Command 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 SCPI command, it cannot be sent with the command. It is usually used as a supplementary description of command parameters.

- **Braces { }**

  The braces usually contain multiple optional parameters; one of the parameters should be selected when sending the command.

  For example, :DISPlay:GRID:MODE { FULL | GRID | CROSS | NONE} .

- **Vertical Bar |**

  The vertical bar is used to separate multiple parameters; one of the parameters should be selected when sending the command.

  For example,:DISPlay:GRID:MODE { FULL | GRID | CROSS | NONE}.

- **Square Brackets [ ]**

  The contents in square brackets (command keywords) can be omissible. If the parameter is omitted, the instrument will set the parameter as the default value.

  For example, for the command :MEASure:NDUTy? [<source>], [<source>] indicates the current channel.

- **Triangular Brackets < >**

  The parameter in the brackets must be replaced with valid value.

  For example, send the command :DISPlay:GRID:BRIGhtness <count> in the format of :DISPlay:GRID:BRIGhtness 30.

# Parameter Description

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

- **Boolean**

  The parameter can be set to ON (1) or OFF (0).

  For example, :SYSTem:LOCK {{1 | ON} | {0 | OFF}}

- **Integer**

  Unless otherwise specified, the parameter can take any valid integer value.

  Note: Do not set decimal as parameter, otherwise, errors may occur.

  For example, <count> in the command :DISPlay:GRID:BRIGhtness <count> can take any integer value from 0 to 100.

- **Real**

  Unless otherwise specified, the parameter can take any valid integer value.

  For example, for C1, <offset> in the command CHANnel1:OFFSet <offset> can take a real number as its value.

- **Discrete**

  The parameter can only take specified numbers or characters.

  For example, the parameter in the command :DISPlay:GRID:MODE { FULL | GRID | CROSS | NONE } can only be FULL, GRID, CROSS, or NONE.

- **ASCII String**

  A string parameter can contain any ASCII character. Strings must begin and end with paired quotation marks, which can be either single or double quotes. To include a quotation mark or delimiter within the string, type it twice without adding any other characters.

  For example, IP: SYST:COMM:LAN:IPAD 192.168.1.10.

# Abbreviation

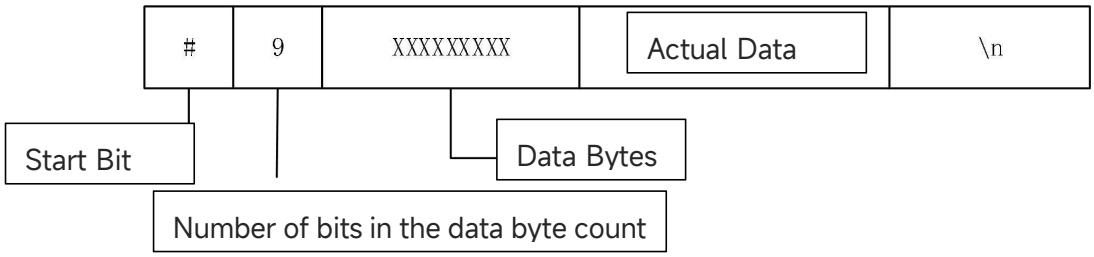
All commands are case-sensitive. If a command is written in an abbreviated format, all capital letters in the command must be input completely.

# Date Return

The data return is divided into single data and batch data returns and must end with a newline <NL> character. If the data is of string type, return the string; if the data is of integer type, return the integer; if the data is of real numeric type, return it in scientific notation, with the part before 'e' retaining the actual significant digits after the decimal point, and the part after 'e' retaining three digits. The batch data return format is data block header + data block, where the data block header has the following format: the first digit (9) after “#” indicates the remaining number of digits in the data block header; the remaining digits in the data block header indicate the number of bytes of data to be transmitted this time (if less than 9 digits, pad with zeros in front). For example, the header for sending a 1000-byte data block is #9000001000.

### Data Block Format

DATA is a data stream, while others are in ASCII strings, represented as <#9XXXXXXXXX + DATA + \n>, as shown in the following figure.




Note: When returning invalid data, use “*” to indicate ASCII type; for real numeric type, return the maximum value that can be represented.

# SCPI

## IEEE488.2 Common Command

### *IDN?

- **Command Format**

  *IDN?

- **Functional Description**

  Query the manufacturer's name, oscilloscope model, product serial number, and software version.

- **Return Format**

  Manufacture name, oscilloscope model, product serial number, and software version are separated by a dot.

- **For Example**

  UNI-T Technologies, MSO1000HD, 123456789, 00.00.01

### *RST

- **Command Format**

  *RST

- **Functional Description**

  Restore factory settings and clear the entire error message, send, and receive queue buffers.

### *OPC

- **Command Format**

  *OPC

  *OPC?

- **Functional Description**

  Query whether the last instruction has been executed or force the execution completion flag to be set to 1.

- **Return Format**

  The query returns whether the last command was executed. 1 indicates that the command was executed, while 0 indicates that the command was not executed.

- **For Example**

| | |
|---|---|
| *OPC | Set the completion flag of the instruction to 1. |
| *OPC? | The query returns 1, indicating that the command was executed; |

otherwise, it was not executed.

# SYSTem Command

The command is used for basic operation of the oscilloscope, including operating control, full keyboard lock, error queue, and system data setting.

### :RUN

- **Command Format**

  :RUN

- **Functional Description**

  Start the sampling operation of the oscilloscope, executing the command “:STOP” to stop the operation.

### :STOP

- **Command Format**

  :STOP

- **Functional Description**

  Stop the sampling operation of the oscilloscope, executing the command “:RUN” to restart the operation.

### :AUTO

- **Command Format**

  :AUTO

- **Functional Description**

  Automatically adjust the instrument's control values, enabling the waveform to achieve the best display effect.

### :SYSTem:LOCK

- **Command Format**

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

  :SYSTem:LOCK?

- **Functional Description**

  Lock or unlock the system; if there is a touch function, it will be also locked.

- **Return Format**

The query returns the system lock status: 0 for unlocked and 1 for locked.

- **For Example**

| | |
|---|---|
| :SYSTem:LOCK ON/:SYST:LOCK 1 | The system is locked. |
| :SYSTem:LOCK OFF/:SYST:LOCK 0 | The system is unlocked. |
| :SYSTem:LOCK? | The query returns 1, indicating that it's locked. |

### :SYSTem:TOUCh:LOCK

- **Command Format**

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

:SYSTem:TOUCh:LOCK?

- **Functional Description**

Lock or unlock the touch function.

- **Return Format**

The query returns the touch function lock state: 0 indicates it's unlocked, while 1 indicates it's locked.

- **For Example**

| | |
|---|---|
| :SYSTem:TOUCh:LOCK ON | The touch function is locked. |
| :SYSTem:TOUCh:LOCK? | The query returns 1, indicating that the touch function is locked. |

### :SYSTem:ERRor

- **Command Format**

:SYSTem:ERRor

:SYSTem:ERRor?

- **Functional Description**

Empty error message queue.

- **Return Format**

The query returns the most recent error message, and the query returns error messages in the format <Message number>, <Message content>. <Message number> is an integer, and <Message content> is an ASCII character string enclosed in double quotation marks.

- **For Example**

| | |
|---|---|
| :SYSTem:ERR | Empty error message queue. |
| :SYSTem:ERR? | The query returns: -113, indicating "Undefined header; command cannot be found." |

### :SYSTem:SETup

- **Command Format**

  :SYSTem:SETup <setup_data>

  :SYSTem:SETup?

- **Functional Description**

  Read or restore the system setting data, <setup_data> conforms to the Data Block Format.

- **Return Format**

  The query returns the system setting data.

### :SYSTem:VERSion?

- **Command Format**

  :SYSTem:VERSion?

- **Return Format**

  The query returns the version information as character string.

  HW represents the hardware version number. SW represents the software number. PD represents the production data. ICV represents the protocol version number.

- **For Example**

  :SYST:VERS? The query returns H.W:1.0;SW:1.0;PD.

### :SYSTem:RTC

- **Command Format**

  :SYSTem:RTC <year>,<month>,<day>,<hour>,<minute>,<second>

  :SYSTem:RTC?

- **Functional Description**

  Set the system time.

- **Return Format**

  The query returns year, month, day, hour, second, and minute.

- **For Example**

  :SYSTem:RTC 2017,7,7,20,8,8 Set the system time to 20:08:08 on July 7th, 2017.

  :SYSTem:RTC? The query returns 2017, 7, 7, 20, 8, 8.

### :SYSTem:CAL

- **Command Format**

  :SYSTem:CAL

- **Functional Description**

  Set self-calibration of the system. During self-calibration, normal communication is disabled.

### :SYSTem:LA:CAL

- **Command Format**

  :SYSTem:LA:CAL

- **Functional Description**

  Set LA elf-calibration of the system. During self-calibration, normal communication is disabled.

### :SYSTem:CLEar

- **Command Format**

  :SYSTem:CLEar

- **Functional Description**

  Clear all saved waveforms and configuration data from the system.

### :SYSTem:BOOT

- **Command Format**

  :SYSTem:BOOT {OPEN|CLOSe|LPOut}

  :SYSTem:BOOT?

- **Functional Description**

  Set the power-up state of the system.

  OPEN: Always on

  CLOSe: Always off

  LPOut: The last power-off state

- **Return Format**

  The query returns the power-up state of the system.

- **For Example**

| | |
|---|---|
| :SYSTem:BOOT CLOSe | Set the power-up state to CLOSe (Always off). |
| :SYSTem:BOOT? | The query returns CLOSe. |

### :SYSTem:BEEP

- **Command Format**

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

  :SYSTem:BEEP?

- **Functional Description**

  Set and query the beep state of the system.

- **Return Format**

  The query returns the beep state of the system: 0 indicates OFF, while 1 indicates ON.

- **For Example**

| | |
|---|---|
| :SYSTem:BEEP ON | Enable the beep. |
| :SYSTem:BEEP? | The query returns 1, indicating that the beep is turned on. |

### :SYSTem:LANGuage

- **Command Format**

  :SYSTem:LANGuage { ENGLish | SIMPlifiedchinese | TRADitionalchinese}

  :SYSTem:LANGuage?

- **Functional Description**

  Set the system language.

- **Return Format**

  The query returns { ENGLish | SIMPlifiedchinese | TRADitionalchinese}.

- **For Example**

| | |
|---|---|
| :SYSTem:LANGuage ENGL | Set the system language to English. |
| :SYSTem:LANGuage? | The query returns ENGLish. |

### :SYSTem:SIGNal:SYNC

- **Command Format**

  :SYSTem:SIGNal:SYNC {IDLE|INPut|OUTPut}

  :SYSTem:SIGNal:SYNC?

- **Functional Description**

  Set and query the synchronization state of the 10 MHz signal: idle, input, or output.

- **Return Format**

  The query returns the synchronization state of the 10 MHz signal.

- **For Example**

| | |
|---|---|
| :SYSTem:SIGNal:SYNC IDLE | Set the synchronization state of the 10 MHz signal to IDLE. |
| :SYSTem:SIGNal:SYNC? | The query returns IDLE. |

### :SYSTem:SQUare<n>:SELect

- **Command Format**

:SYSTem:SQUare<n>:SELect { 10Hz | 100Hz | 1 kHz | 10 kHz | 100 kHz | 3VREF }

:SYSTem:SQUare<n>:SELect?

- **Functional Description**

Set and query the square wave selection, where the DC signal output of 3 V is only supported by Terminal 1.

<n>: {1} represents Terminal 1, respectively.

- **Return Format**

The query returns { 10Hz | 100Hz | 1 kHz | 10 kHz | 100 kHz | 3VREF }.

- **For Example**

| | |
|---|---|
| :SYSTem:SQUare1:SELect 10Hz | Terminal 1 selects the square waveform output of 10 Hz. |
| :SYSTem:SQUare1:SELect? | The query returns 10 Hz. |

### :SYSTem:OUTPut:SELect

- **Command Format**

:SYSTem:OUTPut:SELect { TRIGger | PF}

:SYSTem:OUTPut:SELect?

- **Functional Description**

Set the output of the [AUX OUT] connector on the rear panel to TRIGger (Trigger) or PF (Pass&Fail).

- **Return Format**

The query returns { TRIGger | PF }.

- **For Example**

| | |
|---|---|
| :SYSTem:OUTPut:SELect TRIG | Select the output to TRIGger (Trigger). |
| :SYSTem:OUTPut:SELect? | The query returns “TRIGger.” |

### :SYSTem:VERTical:EXPand

- **Command Format**

:SYSTem:VERTical:EXPand {GND|CENTer}

:SYSTem:VERTical:EXPand?

- **Functional Description**

Set the vertical expand type for system waveform.

CENTer: Refers to the center of the screen. When the vertical scale is adjusted, the waveform expands or compresses around the center of the screen.

GND: Refers to the zero point of the channel. When the vertical scale is adjusted, the waveform

expands or compresses around the zero position of the channel signal.

- **Return Format**

The query returns the vertical expand type.

- **For Example**

| | |
|---|---|
| :SYSTem:VERTical:EXPand CENTer | Set the vertical expand type to CENTer (Center). |
| :SYSTem:VERTical:EXPand? | The query returns CENTer. |

### :SYSTem:LAN:RESet

- **Command Format**

:SYSTem:LAN:RESet

- **Functional Description**

Immediately reset the current network to the default settings.

- **For Example**

| | |
|---|---|
| :SYSTem:LAN:RESet | Immediately reset the current network to the default settings. |

### :SYSTem:LAN:APPLy

- **Command Format**

:SYSTem:LAN:APPLy

- **Functional Description**

Immediately change and apply the current network settings.

- **For Example**

| | |
|---|---|
| :SYSTem:LAN:APPLy | Immediately change and apply the current network settings. |

### :SYSTem:LAN:GATEway

- **Command Format**

:SYSTem:LAN:GATEway <gateway>

:SYSTem:LAN:GATEway?

- **Functional Description**

Set the default gateway. <gateway> is an ASCII string in the format “xxx.xxx.xxx.xxx.”

- **Return Format**

The query returns the default gateway.

- **For Example**

| | |
|---|---|
| :SYST:LAN:GATE "192.168.1.1" | Set the default gateway to 192.168.1.1. |
| :SYST:LAN:GATE? | The query returns 192.168.1.1. |

### :SYSTem:LAN:SMASK

- **Command Format**

  :SYSTem:LAN:SMASK <submask>

  :SYSTem:LAN:SMASK?

- **Functional Description**

  Set the subnet mask. <submask> is an ASCII string in the format “xxx.xxx.xxx.xxx.”

- **Return Format**

  The query returns the subnet mask.

- **For Example**

| | |
|---|---|
| :SYST:LAN:SMASK "255.255.255.0" | Set the subnet mask to 255.255.255.0. |
| :SYST:LAN:SMASK? | The query returns 255.255.255.0. |

### :SYSTem:LAN:IPADdress

- **Command Format**

  :SYSTem:LAN:IPADdress <ip>

  :SYSTem:LAN:IPADdress?

- **Functional Description**

  Set the IP address. <ip> is an ASCII string in the format “xxx.xxx.xxx.xxx.”

- **Return Format**

  The query returns the IP address.

- **For Example**

| | |
|---|---|
| :SYST:LAN:IPAD "192.168.1.10" | Set the IP address to 192.168.1.10. |
| :SYST:LAN:IPAD? | The query returns 192.168.1.10. |

### :SYSTem:LAN:DHCP

- **Command Format**

  :SYSTem:LAN:DHCP {{1 | ON} | {0 | OFF}}

  :SYSTem:LAN:DHCP?

- **Functional Description**

  Switch the configuration mode to Auto (Automatic IP) or Manual (Manual IP).

- **Return Format**

  The query returns the dynamic configuration mode: 0 indicates Manual (Manual IP), while 1 indicates Auto (Automatic IP).

- **For Example**

| | |
|---|---|
| :SYST:LAN:DHCP ON | Enable IP DHCP. |
| :SYST:LAN:DHCP? | The query returns 1. |

### :SYSTem:LAN:MAC?

- **Command Format**

  :SYSTem:LAN:MAC?

- **Return Format**

  The query returns MAC address.

- **For Example**

| | |
|---|---|
| :SYST:LAN:MAC? | The query returns 00-2A-A0-AA-E0-56. |

### :SYSTem:AUTO:CHANnel

- **Command Format**

  :SYSTem:AUTO:CHANnel {AUTO|KEEP}

  :SYSTem:AUTO:CHANnel?

- **Functional Description**

  Set whether the channel maintains its current state under the automatic setting.

  AUTO indicates that the channel is set automatically according to the preset setting; KEEP indicates that the channel is set automatically while maintaining the current state.

- **Return Format**

  The query returns {AUTO|KEEP}.

- **For Example**

| | |
|---|---|
| :SYSTem:AUTO:CHANnel KEEP | Set the channel state to KEEP under the automatic setting. |
| :SYSTem:AUTO:CHANnel? | The query returns KEEP. |

### :SYSTem:AUTO:ACQuire

- **Command Format**

  :SYSTem:AUTO:ACQuire {AUTO|KEEP}

  :SYSTem:AUTO:ACQuire?

- **Functional Description**

  Set whether the sampling maintains its current state under the automatic setting.

  AUTO indicates that the sampling is set automatically according to the preset setting; KEEP indicates that the sampling is set automatically while maintaining the current state.

- **Return Format**

The query returns {AUTO|KEEP}.

- **For Example**

:SYSTem:AUTO:ACQuire KEEP Set the sampling state to KEEP under the automatic setting.

:SYSTem:AUTO:ACQuire? The query returns KEEP.

### :SYSTem:AUTO:TRIGger

- **Command Format**

:SYSTem:AUTO:TRIGger {AUTO|KEEP}

:SYSTem:AUTO:TRIGger?

- **Functional Description**

Set whether the trigger maintains its current state under the automatic setting.

AUTO indicates that the trigger is set automatically according to the preset setting; KEEP indicates that the trigger is set automatically while maintaining the current state.

- **Return Format**

The query returns {AUTO|KEEP}.

- **For Example**

:SYSTem:AUTO:TRIGger KEEP Set the trigger state to KEEP under the automatic setting.

:SYSTem:AUTO:TRIGger? The query returns KEEP.

### :SYSTem:AUTO:SIGNal

- **Command Format**

:SYSTem:AUTO:SIGNal {AUTO|KEEP}

:SYSTem:AUTO:SIGNal?

- **Functional Description**

Set whether the input signal channel (active channel) maintains its current state under the automatic setting.

AUTO indicates that the active channel is set automatically according to the preset setting; KEEP indicates that the active channel is set automatically while maintaining the current state.

- **Return Format**

The query returns {AUTO|KEEP}.

- **For Example**

:SYSTem:AUTO:SIGNal KEEP Set the active channel to KEEP under the automatic setting.

:SYSTem:AUTO:SIGNal? The query returns KEEP.

### :SYSTem:OPTion:INFo

- **Command Format**

  :SYSTem:OPTion:INFo?

- **Functional Description**

  Query the activation status of all function options in the oscilloscope system: 0 indicates the option is not activated, while 1 indicates the option is activated.

- **Return Format**

  The query returns the activation status of all function options. The option list data is arranged in CSV format. The returned data conforms to Data Block Format.

- **For Example**

  :SYSTem:OPTion:INFo? The query returns the activation status of all function options:

  #9000000196OPTION,

  Type, Active, Time,

  CAN, 0, 160H,

  CANFD, 1, *,

  LIN, 1, *,

  FlexRay, 1, *,

  SENT, 1, *,

  AUDio, 0, 160H,

  MIL STD 1553B, 0, 160H,

  ARINC429, 0, 160H,

  Manchester, 0, 160H,

  PWR, 0, 160H,

  AWG, 0, 160H,

  In which, “#9000000196” is the TMC data block header, followed by the data in the option list. In the data block header, the number following “#9” indicates the number of bytes of valid data that follows. OPTION represents the options, with each piece of data separated by commas and each line of data separated by newline characters.

## CHANnel Command

This command is used Set or query the vertical system, including bandwidth limits, coupling, vertical scale, and vertical offset of the channel.

### :CHANnel<n>:BWLimit

- **Command Format**

  :CHANnel<n>:BWLimit {<bandwidth> | FULL}

  :CHANnel<n>:BWLimit?

- **Functional Description**

  Set the bandwidth limits, FULL indicates that the bandwidth limits are disabled, and the full bandwidth is enabled.

  <bandwidth>: Customized bandwidth limits, ranging from 50 Hz to 200 MHz. This indicates that the bandwidth limits are enabled and adjusted to the specified bandwidth. If the high-frequency component of the DUT signal exceeds this bandwidth, it will be attenuated.

  <n>: {1|2|3|4} represents {C1|CH2|CH3|CH4}, respectively.

- **Return Format**

  The query returns the bandwidth limits.

  The query returns FULL when the bandwidth limits are disabled. When enabled, the query returns the bandwidth limits value in scientific notation, with the unit Hz.

- **For Example**

| | |
|---|---|
| :CHAN1:BWL 20MHz | Enable the bandwidth limits 20 MHz for Channel 1. |
| :CHAN1:BWL? | The query returns 2.000000e+01. |

### :CHANnel<n>:COUPling

- **Command Format**

  :CHANnel<n>:COUPling {DC|AC|GND}

  :CHANnel<n>:COUPling?

- **Functional Description**

  Set the coupling mode for the channel.

  DC: Allows both AC and DC components of the input signal to pass.

  AC: Blocks the DC component of the input signal.

  GND: Cuts off the input signal.

  <n>: {1|2|3|4} represents {C1|CH2|CH3|CH4}, respectively.

- **Return Format**

  Query the current coupling mode (AC, DC, or GND) of the channel.

- **For Example**

| | |
|---|---|
| :CHAN1:COUP DC | Set the coupling mode of Channel 1 to DC. |
| :CHAN1:COUP? | The query returns DC. |

### :CHANnel<n>:DISPlay

- **Command Format**

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

  :CHANnel<n>:DISPlay?

- **Functional Description**

  Switch the specified channel to ON or OFF.

  <n>: {1|2|3|4|5|6|7|8} indicates {C1|CH2|CH3|CH4|REFA|REFB|REFC|REFD}, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :CHAN1:DISP ON Enable Channel 1.

  :CHAN1:DISP? The query returns 1, indicating that Channel 1 is enabled.

### :CHANnel<n>:INVert

- **Command Format**

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

  :CHANnel<n>:INVert?

- **Functional Description**

  Switch the waveform inverse phase to ON (enable the waveform inverse phase) or OFF (normal waveform display).

  <n>: {1|2|3|4} represents {C1|CH2|CH3|CH4}, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :CHAN1:INV OFF Disable the inverse phase of Channel 1.

  :CHAN1:INV? The query returns 0, indicating that the inverse phase of Channel 1 is disabled.

### :CHANnel<n>:PROBe

- **Command Format**

  :CHANnel<n>:PROBe { <probe> | 0.001X | 0.01X | 0.1X | 1X | 10X | 100X | 1000X}

  :CHANnel<n>:PROBe?

- **Functional Description**

  Set the probe attenuation ratio, ranging from 0.001X to 20,000X.

  <probe>: The probe attenuation ratio can be set within the specified range.

<n>: {1|2|3|4} represents {C1|CH2|CH3|CH4}, respectively.

- **Return Format**

The query returns the current probe attenuation ratio in scientific notation with the unit in X, when the oscilloscope is set to a continuous probe attenuation ratio.

- **For Example**

| | |
|---|---|
| :CHAN1:PROB 10X | Set the probe attenuation ratio of Channel 1 to 10. |
| :CHAN1:PROB? | The query returns 1.000000e+01. |

### :CHANnel<n>:OFFSet

- **Command Format**

:CHANnel<n>:OFFSet <offset>

:CHANnel<n>:OFFSet?

- **Functional Description**

Set the waveform offset in the vertical direction.

<n>: {1|2|3|4|5|6|7|8} indicates {C1|CH2|CH3|CH4|REFA|REFB|REFC|REFD}, respectively.

- **Return Format**

The query returns the <offset> value in scientific notation, with the unit related to :CHANnel<n>:UNITs.

- **For Example**

| | |
|---|---|
| :CHAN1:OFFS 20V | Set the vertical offset of Channel 1 to 20 V. |
| :CHAN1:OFFS? | The query returns 2.000000e+01. |

### :CHANnel<n>:SCALe

- **Command Format**

:CHANnel<n>:SCALe {<scale> | UP | DOWN}

:CHANnel<n>:SCALe?

- **Functional Description**

Set the volts/div scale in the vertical direction.

<scale>: Volts/div

UP: Increase by one scale based on the current scale.

DOWN : Decrease by one scale based on the current scale.

<n>: {1|2|3|4|5|6|7|8} indicates {C1|CH2|CH3|CH4|REFA|REFB|REFC|REFD}, respectively.

- **Return Format**

The query returns the current volts/div scale in scientific notation, with the unit related

to :CHANnel<n>:UNITs.

- **For Example**

| | |
|---|---|
| :CHAN1:SCAL 20V | Set the volts/div scale of Channel 1 to 20 V. |
| :CHAN1:SCAL? | The query returns 2.000000e+01. |
| :CHAN1:SCAL UP | Increase by one scale based on volts/div scale 20 V. |

### :CHANnel<n>:UNITs

- **Command Format**

:CHANnel<n>:UNITs {VOLTs|AMPeres|WATTs|UNKNown}

:CHANnel<n>:UNITs?

- **Functional Description**

Set the channel's unit to VOLTs (Voltage), AMPeres (Current), WATTs (Power), or UNKNown (Unknown).

<n>: {1|2|3|4} represents {C1|CH2|CH3|CH4}, respectively.

- **Return Format**

The query returns VOLTs (Voltage), AMPeres (Current), WATTs (Power), or UNKNown (Unknown).

- **For Example**

| | |
|---|---|
| :CHAN1:UNIT VOLT | Set Channel 1 unit to VOLTs (Voltage). |
| :CHAN1:UNIT? | The query returns VOLTs. |

### :CHANnel<n>:BIASV

- **Command Format**

:CHANnel<n>:BIASV <value>

:CHANnel<n>:BIASV?

- **Functional Description**

Set the bias value of the specified channel.

<n>: {1|2|3|4} represents {C1|CH2|CH3|CH4}, respectively.

- **Return Format**

The query returns the bias value of the specified channel in scientific notation, with the unit related to :CHANnel<n>:UNITs.

- **For Example**

| | |
|---|---|
| :CHAN1:BIASV 2 | Set the bias value of Channel 1 to 2 V. |
| :CHAN1:BIASV? | The query returns 2.000000e+00. |

### :CHANnel<n>:BIASV:ZERO

- **Command Format**

  :CHANnel<n>:BIASV:ZERO

- **Functional Description**

  Set the bias value of the specified channel to zero.

  <n>: {1|2|3|4} represents {C1|CH2|CH3|CH4}, respectively.

- **For Example**

  :CHAN1:BIASV:ZERO Set the bias value of Channel 1 to zero.

### :CHANnel<n>:VERNier

- **Command Format**

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

  :CHANnel<n>:VERNier?

- **Functional Description**

  Set the scale method to ON or OFF: ON indicates fine tuning for further adjusting the vertical resolution; OFF indicates coarse tuning Adjust the vertical sensitivity with using 1-2-5 system.

  <n>: {1|2|3|4 indicates C1|CH2| CH3|CH4}, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :CHAN1:VERN ON Enable the fine tuning for Channel 1.

  :CHAN1:VERN? The query returns 1.

### :CHANnel<n>:LABel:ENABle

- **Command Format**

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

  :CHANnel<n>:LABel:ENABle?

- **Functional Description**

  Set or query whether the label of the specified channel is enabled.

  <n>: {1|2|3|4|5|6|7|8} indicates {C1|CH2|CH3|CH4|REFA|REFB|REFC|REFD}, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :CHANnel1:LABel:ENABle ON Enable Channel 1 label.

| | |
|---|---|
| :CHANnel1:LABel:ENABle? | The query returns 1, indicating that Channel 1 label is enabled. |

### :CHANnel<n>:LABel

- **Command Format**

  :CHANnel<n>:LABel <label>

  :CHANnel<n>:LABel?

- **Functional Description**

  Set or query the label of the specified channel.

  <n>: {1|2|3|4|5|6|7|8} indicates {C1|CH2|CH3|CH4|REFA|REFB|REFC|REFD}, respectively.

  <label>: ASCII string includes English letters, numbers, and some punctuation marks.

- **Return Format**

  The query returns the label of the physical channel as an ASCII string.

- **For Example**

| | |
|---|---|
| :CHANnel1:LABel "C1" | Set the label of Channel 1 to C1. |
| :CHANnel1:LABel? | The query returns C1. |

### :CHANnel<n>:HIDe

- **Command Format**

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

  :CHANnel<n>:HIDe?

- **Functional Description**

  Set or query the display status of the waveform for the specified channel.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :CHANnel1:HIDe ON | The waveform of the specified channel is hidden. |
| :CHANnel1:HIDe? | The query returns 1, indicating that C1's waveform is hidden. |

### :CHANnel<n>:SELect

- **Command Format**

  :CHANnel<n>:SELect

  :CHANnel<n>:SELect?

- **Functional Description**

  Select a channel.

<n>: {1|2|3|4|5|6|7|8} indicates {C1|CH2|CH3|CH4|REFA|REFB|REFC|REFD}, respectively.

- **Return Format**

  The query returns either 1 or 0.

- **For Example**

| | |
|---|---|
| :CHAN1:SELect | Select Channel 1. |
| :CHAN1:SELect? | The query returns 1, indicating that Channel 1 is selected. |

# TIMebase Command

This command is used to change the horizontal scale (time base) of the current channel and change the horizontal position (trigger offset) of the trigger in memory. Adjusting the horizontal scale will expand or compress the waveform relative to the center of the screen, while changing the horizontal position will shift the waveform relative to the center of the screen.

### :TIMebase:TYPe

- **Command Format**

  :TIMebase:TYPe {XY|YT}

  :TIMebase:TYPe?

- **Functional Description**

  Set the time base type.

  XY: In XY mode, both the X and Y axes represent voltage. This mode is used to detect phase changes when a signal passes through a circuit.

  YT: The X-axis represents horizontal time, while the Y-axis represents vertical voltage.

- **Return Format**

  The query returns {XY|YT}.

- **For Example**

| | |
|---|---|
| :TIMebase:TYPe YT | Set the time base type to YT. |
| :TIMebase:TYPe? | The query returns YT. |

### :TIMebase:XY

- **Command Format**

  :TIMebase:XY <source1>,<source2>

  :TIMebase:XY?

- **Functional Description**

  Set or query the channel relative to the axes in XY mode.

<source1>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

<source2>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **Return Format**

The query returns the channel relative to the axes in XY mode.

- **For Example**

| | |
|---|---|
| :TIMebase:XY CHANnel1,CHANnel2 | Set the channel relative to Channel 1 and Channel 2 in XY mode. |
| :TIMebase:XY? | The query returns CHANnel1, CHANnel2. |

### :TIMebase:EXTend:ENABle

- **Command Format**

:TIMebase:EXTend:ENABle { {1|ON} | {0|OFF} }

:TIMebase:EXTend:ENABle?

- **Functional Description**

Set and query the switch state of the expand time base. If expand time base is disabled, it operates only in main time base mode.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :TIMebase:EXTend:ENABle ON | Enable the expand time base. |
| :TIMebase:EXTend:ENABle? | The query returns 1, indicating that the extended time base is enabled. |

### :TIMebase:OFFSet

- **Command Format**

:TIMebase:OFFSet <offset>

:TIMebase:OFFSet?

- **Functional Description**

Adjust the MAIN time base offset, which changes the waveform position offset relative to the center of the screen.

- **Return Format**

The query returns <offset> value in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TIM:OFFS 1s | Set the MAIN time base offset to 1s. |
| :TIM:OFFS? | The query returns 1.000000e+00. |

### :TIMebase:SCALe

- **Command Format**

  :TIMebase:SCALe {<scale> | UP | DOWN}

  :TIMebase:SCALe?

- **Functional Description**

  Set the scale of the MAIN time base, which is s/div (seconds per division).

  <scale>: the scale of the time base

  UP: Increase by one scale based on the current scale.

  DOWN : Decrease by one scale based on the current scale.

- **Return Format**

  The query returns < scale > value in scientific notation, with the unit in seconds (s)/div.

- **For Example**

| | |
|---|---|
| :TIM:SCAL 2 | Set the offset of the MAIN time base to 2 s/div. |
| :TIM:SCAL? | The query returns 2.000000e+00. |

### :TIMebase:VERNier

- **Command Format**

  :TIMebase:VERNier { {1|ON} | {0|OFF} }

  :TIMebase:VERNier?

- **Functional Description**

  Set the adjustment mode of the scale.

  Fine tuning: When set to ON (enabled), the mode is fine tuning. Fine tuning subdivides the range set by Coarse tuning to enhance horizontal resolution.

  Coarse tuning: When set to OFF (disabled), the mode is coarse tuning. Coarse tuning determines the horizontal sensitivity based on the 1-2-5 steps.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :TIMebase:VERNier ON | Enable fine tuning function. |
| :TIMebase:VERNier? | The query returns 1. |

### :TIMebase:EXTend:AREa

- **Command Format**

  :TIMebase:EXTend:AREa <hp1>,<vp1>,<hp2>,<vp2>

:TIMebase:EXTend:AREa?

- **Functional Description**

Set or query the range of the extended window, use the coordinates of the upper left and bottom right corners to define the screen range. The left boundary should be less than the right boundary, and the upper boundary should be greater than the lower boundary.

<hp1>: Represents the horizontal time value of the upper left point of the area, with the unit in seconds (s).

<vp1>: Represents the channel vertical value of the upper left point of the area. The unit is determined by the channel's vertical unit.

<hp2>: Represents the horizontal time value of the bottom right point of the area, with the unit in seconds (s).

<vp2>: Represents the channel vertical value of the bottom right point of the area. The unit is determined by the channel's vertical unit.

- **Return Format**

The query returns the coordinate value in scientific notation.

- **For Example**

:TIMebase:EXTend:AREa -5us,200mv,5us,-200mv

The extended range is from the upper left point [-5 us, 200 mv] to the bottom right point [5 us, -200 mv].

:TIMebase:EXTend:AREa?

The query returns -5.000000e-06, 2.000000e-01, 5.000000e-06, -2.000000e-01.

### :TIMebase:EXTend:X:SCALe

- **Command Format**

:TIMebase:EXTend:X:SCALe {<scale> | UP | DOWN}

:TIMebase:EXTend:X:SCALe?

- **Functional Description**

Set the zoomed ratio of the time base in the extended (<Zoomed>) display.

<scale>: Zoomed ratio

UP: Increase by one scale based on the current ratio.

DOWN: Decrease by one scale based on the current ratio.

- **Return Format**

The query returns <scale> value in scientific notation, with the unit in X.

- **For Example**

| | |
|---|---|
| :TIM:EXT:X:SCAL 2 | Set the zoomed ratio of the time base to 2X. |
| :TIM:EXT:X:SCAL? | The query returns 2.000000e+00. |

### :TIMebase:EXTend:Y:SCALe

- **Command Format**

  :TIMebase:EXTend:Y:SCALe {<scale> | UP | DOWN}

  :TIMebase:EXTend:Y:SCALe?

- **Functional Description**

  Set the zoomed ratio of the volts/div scale in the extended (<Zoomed>) display.

  <scale>: Zoomed ratio

  UP: Increase by one scale based on the current ratio.

  DOWN: Decrease by one scale based on the current ratio.

- **Return Format**

  The query returns <scale> value in scientific notation, with the unit in X.

- **For Example**

| | |
|---|---|
| :TIM:EXT:Y:SCAL 2 | Set the zoomed ratio of the volts/div scale to 2X. |
| :TIM:EXT:Y:SCAL? | The query returns 2.000000e+00. |

### :TIMebase:ROLL

- **Command Format**

  :TIMebase:ROLL { {1|ON} | {0|OFF} }

  :TIMebase:ROLL?

- **Functional Description**

  Set or query the state of ROLL time base.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :TIMebase:ROLL ON | Enable automatic ROLL mode. |
| :TIMebase:ROLL? | The query returns 1, indicating that automatic ROLL mode is enabled. |

### :TIMebase:HREFerence:MODe

- **Command Format**

  :TIMebase:HREFerence:MODe {CENTer|LB|RB|TRIGger}

  :TIMebase:HREFerence:MODe?

- **Functional Description**

  Set or query the horizontal reference mode when changing the horizontal time base.

  CENTer: The oscilloscope will horizontally expand or compress the waveform around the screen's center.

  LB: The oscilloscope will expand or compress the waveform around the screen's left side.

  RB: The oscilloscope will expand or compress the waveform around the screen's right side.

  TRIGger: The oscilloscope will expand or compress the waveform around the trigger position.

- **Return Format**

  The query returns {CENTer|LB|RB|TRIGger}.

- **For Example**

| | |
|---|---|
| :TIMebase:HREFerence:MODe TRIGger | Set the horizontal reference mode to TRIGger. |
| :TIMebase:HREFerence:MODe? | The query returns TRIGger. |

# MATH Command

This command is used Set various math operation functions for C1, CH2, CH3, and CH4. The operations include addition, subtraction, multiplication, division, FFT, digital filtering, and function expression.

### :MATH<n>:DISPlay

- **Command Format**

  :MATH<n>:DISPlay { {1|ON} | {0|OFF} }

  :MATH<n>:DISPlay?

- **Functional Description**

  Switch the specified Math channel to ON or OFF.

  <n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :MATH1:DISP ON | Enable MATH1. |
| :MATH1:DISP? | The query returns 1, indicating that MATH1 is enabled. |

### :MATH<n>:SCALe

- **Command Format**

:MATH<n>:SCALe {<scale> | UP | DOWN}

:MATH<n>:SCALe?

- **Functional Description**

Set the volts/div scale of MATH waveform in the vertical direction.

<scale>: Volts/div scale value

UP: Increase by one scale based on the current scale.

DOWN : Decrease by one scale based on the current scale.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

The query returns the current volts/div scale value in scientific notation, with the unit related to :CHANnel<n>:UNITs.

- **For Example**

| | |
|---|---|
| :MATH1:SCAL 20V | Set the volts/div scale of Channel 1 to 20 V. |
| :MATH1:SCAL? | The query returns 2.000000e+01. |
| :MATH1:SCAL UP | Increased by one scale based on the volts/div scale 20 V. |

### :MATH<n>:OFFSet

- **Command Format**

:MATH<n>:OFFSet <offset>

:MATH<n>:OFFSet?

- **Functional Description**

Set the offset of MATH waveform in the vertical direction.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

The query returns <offset> value in scientific notation, with the unit related to :CHANnel<n>:UNITs.

- **For Example**

| | |
|---|---|
| :MATH1:OFFS 20V | Set the offset of Channel 1 to 20 V. |
| :MATH1:OFFS? | The query returns 2.000000e+01. |

### :MATH<n>:MODe

- **Command Format**

MATH<n>:MODe {BASic|FILTer|ADVance}

MATH<n>:MODe?

- **Functional Description**

  Select the MATH mode.

  <n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

  The query returns {BASic|FILTer|ADVance}.

- **For Example**

| | |
|---|---|
| MATH1:MODe BASic | Select the MATH mode to BASic (Basic operation). |
| MATH1:MODe? | The query returns BASic. |

### :MATH<n>:OPERation

- **Command Format**

  :MATH<n>:OPERation {ADD | SUBTract | MULTiply | DIVide }

  :MATH<n>:OPERation?

- **Functional Description**

  Set the operator to addition, subtraction, multiplication, or division.

  <n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

  The query returns {ADD | SUBTract | MULTiply | DIVide }.

- **For Example**

| | |
|---|---|
| :MATH1:OPERation ADD | Use the addition operator: src1+src2 |
| :MATH1:OPERation? | The query returns ADD. |

### :MATH<n>:SOURce<m>

- **Command Format**

  :MATH<n>:SOURce<m> {CHANnel1|CHANnel2|CHANnel3|CHANnel4}

  :MATH<n>:SOURce<m>?

- **Functional Description**

  SOURce <m> represents Source 1 or Source 2, where <m> can take value of 1 and 2.

  <n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

  SOURce1 is used Select the first source for the math function, or it can be the single source for the Filter.

  SOURce2 is used Select the second source for the math function, it cannot be the single source for the Filter.

- **Return Format**

The query returns {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :MATH1:SOUR1 CHAN1 | Set Channel 1 as the first source of MATH1. |
| :MATH1:SOUR1? | The query returns CHANnel1. |
| :MATH1:SOUR2 CHAN2 | Set Channel 2 as the second source of MATH1. |
| :MATH1:SOUR2? | The query returns CHANnel2. |
| :MATH1:OPERation ADD | Adds the source 1 and Source 2 of MATH1. |

### :MATH<n>:FILTer:TYPe

- **Command Format**

:MATH<n>:FILTer:TYPe {LP|HP|BP|BS}

:MATH<n>:FILTer:TYPe?

- **Functional Description**

Set the filter type. LP, HP, BP, and BS represents low-pass filter, high-pass filter, band-pass filter, and band-stop filter, respectively.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

The query returns LP, HP, BP, or BS.

- **For Example**

| | |
|---|---|
| :MATH1:SOUR1 CHAN1 | Set Channel 1 as the source of MATH1. |
| :MATH1:FILT:TYPe BP | Set the digital filter of MATH1 to BP (Band-pass). |
| :MATH1:FILT:TYPe? | The query returns BP. |

### :MATH<n>:FILTer:FREQuency:HIGH

- **Command Format**

:MATH<n>:FILTer:FREQuency:HIGH < freq>

:MATH<n>:FILTer:FREQuency:HIGH?

- **Functional Description**

Set the upper limit of the cut-off frequency for the filter. This is suitable for high-pass filters, band-pass filters, and band-stop filters.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

The query returns 1.000000e+03, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :MATH1:SOUR1 CHAN1 | Set Channel 1 as the source of MATH1. |
| :MATH1:FILT:FREQ:HIGH 1 kHz | Set the upper limit of cut-off frequency for MATH1 filter to 1 kHz. |
| :MATH1:FILT:FREQ:HIGH? | The query returns 1.000000e+03. |

### :MATH<n>:FILTer:FREQuency:LOW

- **Command Format**

  :MATH<n>:FILTer:FREQuency:LOW <freq>

  :MATH<n>:FILTer:FREQuency:LOW?

- **Functional Description**

  Set the lower limit of the cut-off frequency for the filter. This is suitable for low-pass filters, band-pass filters, and band-stop filters.

  <n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

  The query returns 6.000000e+01, with the unit is Hz.

- **For Example**

| | |
|---|---|
| :MATH1:SOUR1 CHAN1 | Set Channel 1 as the source of MATH1. |
| :MATH1:FILT:FREQ:LOW 60Hz | Set the lower limit of cut-off frequency for MATH1 filter to 60 Hz. |
| :MATH1:FILT:FREQ:LOW? | The query returns 6.000000e+01. |

### :MATH<n>:EXPRession

- **Command Format**

  :MATH<n>:EXPRession <expression>

- **Functional Description**

  Use free combination expressions to perform mathematical calculations.

  <n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

  The expression format is in the “Advance” option the of MATH menu. <expression> is an ASCII string.

- **For Example**

  :MATH1:EXPRession "C1*C2" Multiply Channel 1 by Channel 2 of MATH1.

### :MATH<n>:LABel:ENABle

- **Command Format**

:MATH<n>:LABel:ENABle { {1|ON} | {0|OFF} }

:MATH<n>:LABel:ENABle?

- **Functional Description**

Set or query whether the specified channel is enabled.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :MATH1:LABel:ENABle ON | Enable MATH1's label. |
| :MATH1:LABel:ENABle? | The query returns 1, indicating that MATH1's label is enabled. |

### :MATH<n>:LABel

- **Command Format**

:MATH<n>:LABel <label>

:MATH<n>:LABel?

- **Functional Description**

Set or query the label of the specified math channel.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

<label>: An ASCII string includes English letters, numbers, and some punctuation marks.

- **Return Format**

The query returns the label of the specified physical channel in ASCII string format.

- **For Example**

| | |
|---|---|
| :MATH1:LABel "M1" | Set the label of MATH1 to M1. |
| :MATH1:LABel? | The query returns M1. |

### :MATH<n>:UNITs

- **Command Format**

:MATH<n>:UNITs <unit>

:MATH<n>:UNITs?

- **Functional Description**

Set or query the customized unit of the specified math channel.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

<unit>: An ASCII string includes English letters, numbers, and some punctuation marks.

- **Return Format**

The query returns the customized unit of the specified physical channel in ASCII string format.

- **For Example**

| | |
|---|---|
| :MATH1:UNITs "VV" | Set the customized unit of MATH1 to VV. |
| :MATH1:UNITs? | The query returns VV. |

### :MATH<n>:SELect

- **Command Format**

:MATH<n>:SELect

:MATH<n>:SELect?

- **Functional Description**

Select a math channel.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

The query returns either 1 or 0.

- **For Example**

| | |
|---|---|
| :MATH1:SELect | Select MATH1. |
| :MATH1:SELect? | The query returns 1, indicating that MATH1 is selected. |

### :MATH<n>:INDPendent

- **Command Format**

:MATH<n>:INDPendent { {1|ON} | {0|OFF} }

:MATH<n>:INDPendent?

- **Functional Description**

Set the independent window of the specified math channel to ON or OFF.

<n>: {1|2|3|4} represents {MATH1|MATH2|MATH3|MATH4}, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :MATH1:INDPendent ON | Enable the independent window for MATH1. |
| :MATH1:INDPendent? | The query returns 1, indicating that the independent window of MATH1 is enabled. |

## FFT Command

This command is used Set FFT calculations on C1, CH2, CH3, and CH4 waveforms.

### :FFT<n>:DISPlay

- **Command Format**

  :FFT<n>:DISPlay { {1|ON} | {0|OFF} }

  :FFT<n>:DISPlay?

- **Functional Description**

  Set the specified FFT calculations to ON or OFF.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :FFT1:DISP ON | Enable FFT1 calculation. |
| :FFT1:DISP? | The query returns 1, indicating that FFT1 calculation is enabled. |

### :FFT<n>:AUToset

- **Command Format**

  :FFT<n>:AUToset

- **Functional Description**

  Set the automatic setting of the specified FFT spectrum waveform.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **For Example**

| | |
|---|---|
| :FFT1:AUToset | Set the automatic setting of the specified FFT spectrum waveform. |

### :FFT<n>:SCALe

- **Command Format**

  :FFT<n>:SCALe {<scale> | UP | DOWN}

  :FFT<n>:SCALe?

- **Functional Description**

  Set the scale value of FFT waveform in the vertical direction.

  <scale>: Vertical scale value

  UP: Increased by one scale based on the current scale.

  DOWN: Decreased by one scale based on the current scale.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

  The query returns the current volts/div scale value in scientific notation, with the unit related

to :FFT<n>:VTYPe.

- **For Example**

| | |
|---|---|
| :FFT1:SCAL 2mV | Set the vertical scale of FFT1 to 2 mV. |
| :FFT1:SCAL? | The query returns 2.000000e-03. |
| :FFT1:SCAL UP | Increased one scale based on volts/div 2 mV. |

### :FFT<n>:OFFSet

- **Command Format**

:FFT<n>:OFFSet <offset>

:FFT<n>:OFFSet?

- **Functional Description**

Set the offset of FFT waveform in the vertical direction.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns <offset> value in scientific notation, with the unit related to **:FFT<n>:VTYPe.**

- **For Example**

| | |
|---|---|
| :FFT1:OFFS 200uV | Set the vertical offset of FFT1 to 200 uV. |
| :FFT1:OFFS? | The query returns 2.000000e-04. |

### :FFT<n>:SOURce

- **Command Format**

:FFT<n>:SOURce {CHANnel1|CHANnel2|CHANnel3|CHANnel4}

:FFT<n>:SOURce?

- **Functional Description**

Set the operation source for the specified FFT.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :FFT1:SOUR CHAN1 | Set FFT1 source as Channel 1. |

### :FFT<n>:WINDow

- **Command Format**

:FFT<n>:WINDow {RECTangular|HANNing|HAMMing|BMAN}

:FFT<n>:WINDow?

- **Functional Description**

FFT adds a windowing function to capture a signal. RECT, HANN, HAMM, and BMAN represents Rectangle, Hanning, Hamming, and Blackman window functions, respectively.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns {RECTangular|HANNing|HAMMing|BMAN}.

- **For Example**

| | |
|---|---|
| :FFT1:SOUR CHAN1 | Set FFT1 source as Channel 1. |
| :FFT1:WIND HAMM | Adds Hamming windowing function. |
| :FFT1:WIND? | The query returns HAMMing. |

### :FFT<n>:POINts

- **Command Format**

:FFT<n>:POINts {8K|16K|32K|64K|128K|256K|512K|1M|2M|4M}

:FFT<n>:POINts?

- **Functional Description**

Set the point for the specified FFT calculation.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns {8K|16K|32K|64K|128K|256K|512K|1M|2M|4M}.

- **For Example**

| | |
|---|---|
| :FFT1:POINts 8K | Set FFT1 point to 8K. |
| :FFT1:POINts? | The query returns 8K. |

### :FFT<n>:VTYPe

- **Command Format**

:FFT<n>:VTYPe{VRMS|DBRMS}

:FFT<n>:VTYPe?

- **Functional Description**

Select the unit of FFT calculations in the vertical direction as dBRMS or VRMS.

dBRMS represents power RMS. VRMS represents voltage RMS.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns VRMS or DBRMS.

- **For Example**

:FFT1:SOUR CHAN1 Set Channel 1 as the source.

:FFT1:VTYP VRMS Set the unit of FFT1 calculation in the vertical direction as VRMS.

:FFT1:VTYP? The query returns VRMS.

### :FFT<n>:WATerfall

- **Command Format**

:FFT<n>:WATerfall { {1|ON} | {0|OFF} }

:FFT<n>:WATerfall?

- **Functional Description**

Set the waterfall curve of the specified FFT.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

:FFT1:WATerfall ON Enable the waterfall curve for FFT1.

:FFT1:WATerfall? The query returns 1, indicating that the waterfall curve of FFT1 is enabled.

### :FFT<n>:FREQuency:MODe

- **Command Format**

:FFT<n>:FREQuency:MODe {SPAN | RANG}

:FFT<n>:FREQuency:MODe?

- **Functional Description**

Set the frequency mode for the specified FFT calculation.

In SPAN mode, the center frequency and bandwidth can be set. In RANG mode, the start and stop frequency can be set.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns {SPAN | RANG}.

- **For Example**

:FFT1:FREQuency:MODe SPAN Set the frequency mode of FFT1 to SPAN (Band width).

:FFT1:FREQuency:MODe? The query returns SPAN.

### :FFT<n>:FREQuency:SPAN

- **Command Format**

  :FFT<n>:FREQuency:SPAN 

  :FFT<n>:FREQuency:SPAN?

- **Functional Description**

  Set the bandwidth for the specified FFT.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  : Frequency bandwidth

- **Return Format**

  The query returns the bandwidth of the specified FFT, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :FFT1:FREQuency:SPAN 1 kHz | Set the frequency bandwidth of FFT1 to 1 kHz. |
| :FFT1:FREQ:SPAN? | The query returns 1.000000e+03. |

### :FFT<n>:FREQuency:CENTer

- **Command Format**

  :FFT<n>:FREQuency:CENTer <freq>

  :FFT<n>:FREQuency:CENTer?

- **Functional Description**

  Set the center frequency for the specified FFT.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

  The query returns the center frequency of the specified FFT, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :FFT1:FREQuency:CENTer 1 kHz | Set the center frequency of FFT1 to 1 kHz. |
| :FFT1:FREQ:CENTer? | The query returns 1.000000e+03. |

### :FFT<n>:FREQuency:STARt

- **Command Format**

  :FFT<n>:FREQuency:STARt <freq>

  :FFT<n>:FREQuency:STARt?

- **Functional Description**

  Set the start frequency for the specified FFT.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

  The query returns the start frequency of the specified FFT, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :FFT1:FREQuency:STARt 1 kHz | Set the start frequency of FFT1 to 1 kHz. |
| :FFT1:FREQ:STARt? | The query returns 1.000000e+03. |

### :FFT<n>:FREQuency:STOP

- **Command Format**

  :FFT<n>:FREQuency:STOP <freq>

  :FFT<n>:FREQuency:STOP?

- **Functional Description**

  Set the stop frequency for the specified FFT.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

  The query returns the stop frequency of the specified FFT, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :FFT1:FREQuency:STOP 1 kHz | Set the stop frequency of FFT1 to 1 kHz. |
| :FFT1:FREQ:STOP? | The query returns 1.000000e+03. |

### :FFT<n>:DETEction:REALTime

- **Command Format**

  :FFT<n>:DETEction:REALTime {PPEAK|NPEAK|AVERage|SAMPle}

  :FFT<n>:DETEction:REALTime?

- **Functional Description**

  Set the detection mode for real-time spectrum frequency.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  PPEAK: Takes the maximum of each sampling interval.

  NPEAK: Takes the minimum of each sampling interval.

  AVERage: Takes the average of each sampling interval.

  SAMPle: Takes the first point value of each sampling interval.

- **Return Format**

  The query returns the detection mode of real-time spectrum frequency.

- **For Example**

| | |
|---|---|
| :FFT1:DETEction:REALTime PPEAK | Set the detection mode for real-time spectrum |

| | |
|---|---|
| | frequency of FFT1 to PPEAK (Peak to peak). |
| :FFT1:DETEction:REALTime? | The query returns PPEAK. |

### :FFT<n>:DETEction:AVERage

- **Command Format**

  :FFT<n>:DETEction:AVERage {OFF|PPEAK|NPEAK|AVERage|SAMPle}

  :FFT<n>:DETEction:AVERage?

- **Functional Description**

  Set the detection mode for average spectrum frequency.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  OFF: Disables average spectrum frequency.

  PPEAK: Takes the maximum of each sampling interval.

  NPEAK: Takes the minimum of each sampling interval.

  AVERage: Takes the average of each sampling interval.

  SAMPle: Takes the first point value of each sampling interval.

- **Return Format**

  The query returns the detection mode of average spectrum frequency.

- **For Example**

| | |
|---|---|
| :FFT1:DETEction:AVERage PPEAK | Set the detection mode for the average spectrum frequency of FFT1 to PPEAK (Peak to peak). |
| :FFT1:DETEction:AVERage? | The query returns PPEAK. |

### :FFT<n>:DETEction:AVERage:COUNt

- **Command Format**

  :FFT<n>:DETEction:AVERage:COUNt <value>

  :FFT<n>:DETEction:AVERage:COUNt?

- **Functional Description**

  Set the average count for the average spectrum frequency. The range is 2^n, where n can take a value from 1-10.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

  The query returns the average count of the average spectrum frequency.

- **For Example**

| | |
|---|---|
| :FFT1:DETEction:AVERage:COUNt 64 | Set the average count for the average spectrum |

| | |
|---|---|
| | frequency of FFT1 to 64. |
| :FFT1:DETEction:AVERage:COUNt? | The query returns 64. |

### :FFT<n>:DETEction:MAXHold

- **Command Format**

  :FFT<n>:DETEction:MAXHold {OFF|PPEAK|NPEAK| AVERage|SAMPle}

  :FFT<n>:DETEction:MAXHold?

- **Functional Description**

  Set the detection mode for the maximum hold spectrum frequency.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  OFF: Disables average spectrum frequency.

  PPEAK: Takes the maximum of each sampling interval.

  NPEAK: Takes the minimum of each sampling interval.

  AVERage: Takes the average of each sampling interval.

  SAMPle: Takes the first point value of each sampling interval.

- **Return Format**

  The query returns the detection mode of the maximum hold spectrum frequency.

- **For Example**

| | |
|---|---|
| :FFT1:DETEction:MAXHold PPEAK | Set the detection mode for the maximum hold spectrum frequency to PPEAK (Peak to peak). |
| :FFT1:DETEction:MAXHold? | The query returns PPEAK. |

### :FFT<n>:DETEction:MINHold

- **Command Format**

  :FFT<n>:DETEction:MINHold {OFF|PPEAK|NPEAK| AVERage|SAMPle}

  :FFT<n>:DETEction:MINHold?

- **Functional Description**

  Set the detection mode for the minimum hold spectrum frequency.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  OFF: Disables average spectrum frequency.

  PPEAK: Takes the maximum of each sampling interval.

  NPEAK: Takes the minimum of each sampling interval.

  AVERage: Takes the average of each sampling interval.

  SAMPle: Takes the first point value of each sampling interval.

- **Return Format**

  The query returns the detection mode of the minimum hold spectrum frequency.

- **For Example**

| | |
|---|---|
| :FFT1:DETEction:MINHold PPEAK | Set the detection mode for the minimum hold spectrum frequency to PPEAK (Peak to peak). |
| :FFT1:DETEction:MINHold? | The query returns PPEAK. |

### :FFT<n>:MARKer:SOURce

- **Command Format**

  :FFT<n>:MARKer:SOURce {REALtime|AVERage|MAXHold|MINHold}

  :FFT<n>:MARKer:SOURce?

- **Functional Description**

  Set the mark source for the spectrum frequency marker. This command is commonly used in spectrum frequency.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  REALtime: Marks real-time spectrum frequency.

  AVERage: Marks the average spectrum frequency.

  MAXHold: Marks the maximum hold spectrum frequency.

  MINHold: Marks the minimum hold spectrum frequency.

- **Return Format**

  The query returns the currently selected mark source.

- **For Example**

| | |
|---|---|
| :FFT1:MARKer:SOURce AVERage | Set the mark source of FFT1 to AVERage. |
| :FFT1:MARKer:SOURce? | The query returns AVERage. |

### :FFT<n>:MARKer:TYPe

- **Command Format**

  :FFT<n>:MARKer:TYPe {AUTO|THReshold| MANUal}

  :FFT<n>:MARKer:TYPe?

- **Functional Description**

  Set the mark type for the spectrum frequency.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  AUTO: Automatically mark the spectrum frequency.

  THReshold: Threshold marks the spectrum frequency.

MANUal: Manually mark the spectrum frequency.

- **Return Format**

The query returns the currently selected mark type.

- **For Example**

| | |
|---|---|
| :FFT1:MARKer:TYPe AUTO | Set the mark type of FFT1 to AUTO. |
| :FFT1:MARKer:TYPe? | The query returns AUTO. |

### :FFT<n>:MARKer:POINts

- **Command Format**

:FFT<n>:MARKer:POINts <value>

:FFT<n>:MARKer:POINts ?

- **Functional Description**

Set the mark count for the spectrum frequency marker.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

<value>: Mark count value, ranging from 1-10.

- **Return Format**

The query returns the mark count of the spectrum frequency marker.

- **For Example**

| | |
|---|---|
| :FFT1:MARKer:POINts 10 | Set the mark count of FFT1 marker to 10. |
| :FFT1:MARKer:POINts? | The query returns 10. |

### :FFT<n>:MARKer:EVENt

- **Command Format**

:FFT<n>:MARKer:EVENt {1 | ON} | {0 | OFF}

:FFT<n>:MARKer:EVENt?

- **Functional Description**

Switch the marker list of the spectrum frequency to ON or OFF.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :FFT1:MARKer:EVENt ON | Enable the marker list for FFT1. |
| :FFT1:MARKer:EVENt? | The query returns 1. |

### :FFT<n>:MARKer:DATA?

- **Command Format**

  :FFT<n>:MARKer:DATA?

- **Functional Description**

  Read the mark event data table under FFT.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **Return Format**

  The query returns the mark event data table under FFT. The returned data conforms Data Block Format .

- **For Example**

  :FFT1:MARKer:DATA? The query returns the mark event data of FFT1:

  #9000000089FFT,

  ID,Freq,Amp,

  1, 1.000000e+03, 7.800000e-01,

  2, 2.000000e+03, 7.900000e-01,

  3, 3.000000e+03, 7.700000e-01,

  4, 4.000000e+03, 7.300000e-01,

  5, 5.000000e+03, 7.400000e-01,

  FFT represents FFT mode, with the event table data in CSV format following. The specified format of the event data table is automatically adapted by different devices. The data are separated by commas and will automatically line wrap according to the list.

### :FFT<n>:MARKer:THReshold:LEVel

- **Command Format**

  :FFT<n>:MARKer:THReshold:LEVel <value>

  :FFT<n>:MARKer:THReshold:LEVel?

- **Functional Description**

  Set the threshold voltage for the threshold spectrum frequency marker.

  <n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

  <value>: Threshold voltage. When the vertical unit is Vrms, the unit is V, ranging from 1 mVrms to 100 KVrms; when the vertical unit is dBVrms, the unit is dB, ranging from -60 dB to 100 dB. The setting becomes invalid if the units do not match.

- **Return Format**

  The query returns the threshold of the spectrum frequency marker in scientific notation. The

unit is related to :FFT<n>:VTYPe.

- **For Example**

| | |
|---|---|
| :FFT1:MARKer:THReshold:LEVel -12.5dB | Set the threshold for FFT1 threshold spectrum frequency marker to -12.5 dB. |
| :FFT1:MARKer:THReshold:LEVel? | The query returns -1.250000e-01. |
| :FFT1:MARKer:THReshold:LEVel 0.15V | Set the threshold for FFT1 threshold spectrum frequency marker to 0.15 V. |
| :FFT1:MARKer:THReshold:LEVel? | The query returns 1.500000e-01. |

### :FFT<n>:MARKer:MANUal:PEAK

- **Command Format**

:FFT<n>:MARKer:MANUal:PEAK

- **Functional Description**

Move the marker to the maximum peak.

<n>: {1|2|3|4} represents {FFT1|FFT2|FFT3|FFT4}, respectively.

- **For Example**

| | |
|---|---|
| :FFT1:MARKer:MANUal:PEAK | Manually move the marker to the maximum peak. |

## MEASure Command

Set and query the measurement parameters, including set the measurement statistics and custom measurement functions, for quick waveform analysis. If the current measurement source has no signal input or if the measured result is outside of the valid range (too large or too small), the results are invalid.

**Measurement Parameter Table**

| Type | Parameter | Description | Slave source |
|---|---|---|---|
| Vertical | Maximum (VMAX) | The voltage value of the waveform from its highest point to ground (GND). | |
| Vertical | Minimum (VMIN) | The voltage value of the waveform from its lowest point to ground (GND). | |
| Vertical | Peak to peak (VPP) | The voltage value of the waveforms from its highest point to its lowest point. | |
| Vertical | Top (VTOP) | The voltage value of the waveform from its flat top to ground (GND). | |

| | | | |
|---|---|---|---|
| Vertical | Bottom (VBASe) | The voltage value of the waveform from its bottom to ground (GND). | |
| Vertical | Amplitude (VAMP) | The voltage value of the waveform from its flat top to its bottom. | |
| Vertical | Midpoint (VMID) | Half the sum of the voltage values at the midpoint of the waveform between the top and bottom values. | |
| Vertical | Average (VAVG) | The math average value of the entire waveform or a selected area. | |
| Vertical | Period average (PVAVg) | The average voltage of the waveform points within one period. | |
| Vertical | VRMS | The RMS value over the entire waveform or a selected region is based on the energy generated by the AC signal in the conversion, corresponding to the DC voltage that generates the equivalent energy. | |
| Vertical | Period RMS (PVRMs) | The energy generated by the conversion of the RMS value of the AC signal within one cycle corresponds to the DC voltage that generates the equivalent energy. | |
| Vertical | AC RMS (ACRMs) | Waveform RMS with DC component removed. | |
| Vertical | Period AC RMS (PACRms) | The standard deviation of the voltage value of the waveform data with the DC component removed within a period, represents the RMS measurement of the waveform excluding the DC component. | |
| Vertical | Area (MARea) | The area of the entire waveform displayed on the screen is measured in units of “V*s." Areas above the zero reference (vertical offset) are considered positive, while those below are negative. This measured area is the algebraic sum of the products of voltage and time for all points across the entire waveform shown on the | |

| | | | |
|---|---|---|---|
| | | screen. | |
| Vertical | Perodic area (MPARea) | The area of the first period of the screen waveform is measured in “V*s." Areas above the zero reference (vertical offset) are considered positive, while those below are negative. This measured area is the algebraic sum of the products of voltage and time for all points within the entire one period. | |
| Vertical | Positive area (PMARea) | The algebraic sum of the product of all voltages and times greater than GND (ground) displayed on the screen, measured in "V*s." | |
| Vertical | Negative area (NMARea) | The algebraic sum of the product of all voltages and times less than GND (ground) displayed on the screen, measured in "V*s." | |
| Vertical | Perodic positive area (PMPArea) | The algebraic sum of the products of all voltages greater than GND (ground) and time within the first cycle of the screen waveform, measured in "V*s." | |
| Vertical | Perodic negative area (NMPArea) | The algebraic sum of the products of all voltages are less than GND (ground) and time within the first cycle of the screen waveform, measured in "V*s." | |
| Vertical | Positive overshoot (POVershoot) | The ratio of the difference between the local maximum value and the peak value of the overshoot after the rising edge of the waveform to the amplitude. | |
| Vertical | Negative overshoot (NOVershoot) | The ratio of the difference between the local minimum value and the bottom value of the overshoot after the falling edge of the waveform to the amplitude. | |
| Vertical | Positive preshoot (PPReshoot) | The ratio of the difference between the local minimum value and the bottom value of the preshoot after the | |

| | | | |
|---|---|---|---|
| | | rising edge of the waveform to the amplitude. | |
| Vertical | Negative preshoot (NPReshoot) | The ratio of the difference between the local maximum value and the top value of the preshoot after the falling edge of the waveform to the amplitude. | |
| Horizontal | Period (PERiod) | Defined as the time between the mid-threshold intersections of two consecutive, same polarity edges of a repetitive waveform. | |
| Horizontal | Frequency (FREQuency) | Reciprocal of period | |
| Horizontal | Rise time (RTIMe) | The time needed for the amplitude of the signal waveform to rise from the low value of the lower threshold to the high value of the upper threshold. | |
| Horizontal | Fall time (FTIMe) | The time needed for the amplitude of the signal waveform falls from the high value of the upper threshold to the low value of the lower threshold. | |
| Horizontal | Positive pulse width (PWIDth) | The time difference between the midpoint threshold of the rising edge of the pulse and the midpoint threshold of the following falling edge. | |
| Horizontal | Negative pulse width (NWIDth) | The time difference between the midpoint threshold of the falling edge of the pulse and the midpoint threshold of the following rising edge. | |
| Horizontal | Positive duty ratio (PDUTy) | The ratio of positive pulse width to the period. | |
| Horizontal | Negative duty ratio (NDUTy) | The ratio of the negative pulse width to the period. | |
| Horizontal | Positive pulse number (PPULses) | The number of positive pulses that rise from below the low value of the lower threshold to above the high value of the upper threshold. | |
| Horizontal | Negative pulse number (NPULses) | The number of negative pulses that fall from above the high value of the upper threshold to below the low value of the | |

| | | | |
|---|---|---|---|
| | | lower threshold. | |
| Horizontal | Rising edge number (PEDGes) | The number of rising edges that rise from below the low value of the lower threshold to above the high value of the upper threshold. | |
| Horizontal | Falling edge number (NEDGes) | The number of falling edges that fall from above the high value of the upper threshold to below the low value of the lower threshold. | |
| Horizontal | Burst width (BWIDTh) | The duration of multiple consecutive crossings above the mid-reference level. | |
| Horizontal | Burst interval (BINTerval) | Time interval between two bursts | |
| Horizontal | Burst period (BPERiod) | The burst period when both burst width and burst interval conditions are met. | |
| Horizontal | Burst cycle number (BCYCles) | The number of burst periods when both burst width and burst interval conditions are met. | |
| Other | Ratio (ARRAtio) | The ratio of the RMS voltage of the master source to that of the slave source, expressed in dB. | √ |
| Other | Periodic ratio (ARPRatio) | The ratio of the RMS voltage of the master source to that of the slave source over one cycle, expressed in dB. | √ |
| Other | Setup time (STIMe) | The time from exceeding the specified mid-reference level on the data source to the nearest subsequent exceeding on the clock source's specified mid-reference level. | √ |
| Other | Hold time (HTIMe) | The time from exceeding the specified mid-reference level on the clock source to the nearest subsequent exceeding on the data source's specified mid-reference level. | √ |
| Other | Setup and Hold time (SHRAtio) | The ratio of setup time to total hold time. | √ |
| Other | FRFR (FRFR) | The time difference between the first | √ |

| | | | |
|---|---|---|---|
| | | rising edge of the primary source 1 to the first rising edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold intersections. | |
| Other | FRFF (FRFF) | The time difference between the first rising edge of the primary source 1 to the first rising edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold intersections. | √ |
| Other | FFFR (FFFR) | The time difference between the first falling edge of the primary source 1 to the first rising edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold intersections. | √ |
| Other | FFFF (FFFF) | The time difference between the first falling edge of the primary source 1 to the first falling edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold intersections. | √ |
| Other | FRLF (FRLF) | The time difference between the last rising edge of the primary source 1 to the last falling edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold intersections. | √ |
| Other | FRLR (FRLR) | The time difference between the last rising edge of the primary source 1 to the last rising edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold intersections. | √ |
| Other | FFLR (FFLR) | The time difference between the last falling edge of the primary source 1 to the last rising edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold | √ |

| | | | |
|---|---|---|---|
| | | intersections. | |
| Other | FFLR (FFLF) | The time difference between the last falling edge of the primary source 1 to the last falling edge of the secondary source 2, expressed in seconds, is the time between the midpoint threshold intersections. | √ |
| Other | Phase (r-r) (RRPHase) | The phase difference between the rising edge of the primary source and the rising edge of the secondary source at the midpoint threshold is expressed as a phase shift in degrees. | √ |
| Other | Phase (f-f) (FFPHase) | The phase difference between the rising edge of the primary source and the falling edge of the secondary source at the midpoint threshold is expressed as a phase shift in degrees. | √ |
| Other | Delay (r-r) (RRDElay) | The delay time between the rising edge of the master source and the rising edge of the slave signal source at the median value of the waveform threshold. | √ |
| Other | Delay (f-f) (FFDElay) | The delay time between the falling edge of the master signal source and the falling edge of the slave signal source, measured at the median value of the waveform threshold. | √ |

### :MEASure:CLEar

- **Command Format**

  :MEASure:CLEar

- **Functional Description**

  Clear all currently open measurement items.

- **For Example**

  :MEAS:CLE Clear all currently open measurement items.

### :MEASure:DATA

- **Command Format**

:MEASure:DATA <source>

:MEASure:DATA? <source>

- **Functional Description**

Open the measurement snapshot of the specified source and query all the measured results in the snapshot.

Performs a channel parameter snapshot measurement and returns the data upon sending the query command.

<source> represents the main source:

{CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

The order of measurement snapshot the order is shown below. Refer to Measurement Parameter Table.

{VMAX|VMIN|VPP|VTOP|VBASe|VAMP|VMID|VAVG|PVAVg|VRMS|PVRMs|ACRMs|PACRms|MAREa|MPARea|PMARea|NMARea|PMPArea|NMPArea|POVershoot|NOVershoot|PPReshoot|NPReshoot|PERiod|FREQuency|RTIMe|FTIMe|PWIDth|NWIDth|PDUTy|NDUTy|PPULses|NPULses|PEDGes|NEDGes|BWIDTh|BINTerval|BPERiod|BCYCles}.

- **Return Format**

The query returns the results of channel parameter snapshot measurement. The returned data conforms to the Data Block Format in scientific notation, follows the sequence, and is separated by commas. Invalid values are represented by the maximum real data value.

- **For Example**

| | |
|---|---|
| :MEASure:DATA CHANNel1 | Enable all parameter snapshot measurements for Channel 1. |
| :MEASure:DATA? CHANNel1 | The query returns all parameter snapshot measurements of Channel 1. |

For example, "#90000001001.200000e+00,2.000000e+02,1.200000e+03....."

### :MEASure:ITEM

- **Command Format**

:MEASure:ITEM <item>,<source1>,[<source2>]

:MEASure:ITEM? <item>,<source1>,[<source2>]

- **Functional Description**

Open the arbitrary waveform measurement of the specified source and query the measured results.

Performs an arbitrary waveform measurement and returns the data upon sending the query command.

<item> represents waveform parameter: Refer to Measurement Parameter Table.

{VMAX|VMIN|VPP|VTOP|VBASe|VAMP|VMID|VAVG|PVAVg|VRMS|PVRMs|ACRMs|PACRms|MAREa|MPARea|PMARea|NMARea|PMPArea|NMPArea|POVershoot|NOVershoot|PPReshoot|NPReshoot|PERiod|FREQuency|RTIMe|FTIMe|PWIDth|NWIDth|PDUTy|NDUTy|PPULses|NPULses|PEDGes|NEDGes|BWIDTh|BINTerval|BPERiod|BCYCles|ARRAtio|ARPRatio|STIMe|HTIMe|SHRAtio|FRFR|FRFF|FFFR|FFFF|FRLF|FRLR|FFLR|FFLF|RRPHase|FFPHase|RRDElay|FFDElay}.

<source1> represents the main source:

{CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

<source2> represents the slave source:

{CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

Explanation:

If the measurement parameter is single source, set one source and omit the slave source.

If the measurement parameter requires both sources, set two sources and do not omit the slave source.

- **Return Format**

The query returns the current measured results in scientific notation.

- **For Example**

| | |
|---|---|
| :MEASure:ITEM VAMP,CHANnel1 | Enable and add the amplitude measurement for Channel 1. |
| :MEASure:ITEM? VAMP,CHANnel1 | Perform a measurement, and the query returns 1.233000e-03, with the unit in V. |

### :MEASure:STATistic:DISPlay

- **Command Format**

:MEASure:STATistic:DISPlay {{1|ON}|{0|OFF}}

:MEASure:STATistic:DISPlay?

- **Functional Description**

Switch the measurement statistics to ON or OFF.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :MEASure:STATistic:DISPlay ON | Enable the measurement statistics. |
| :MEASure:STATistic:DISPlay? | The query returns 1. |

### :MEASure:STATistic:UNLimited

- **Command Format**

  :MEASure:STATistic:UNLimited {{1|ON}|{0|OFF}}

  :MEASure:STATistic:UNLimited?

- **Functional Description**

  Switch the unlimited measurement statistics to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :MEASure:STATistic:UNLimited ON | Enable the unlimited measurement statistics. |
| :MEASure:STATistic:UNLimited? | The query returns 1. |

### :MEASure:STATistic:COUNt

- **Command Format**

  :MEASure:STATistic:COUNt <count>

  :MEASure:STATistic:COUNt?

- **Functional Description**

  Set or query the count of the measurement statistics.

  <count> indicates the count of the statistics, expressed as an integer.

- **Return Format**

  The query returns the integer value of the statistic count.

- **For Example**

| | |
|---|---|
| :MEASure:STATistic:COUNt 200 | Set the statistic count to 200. |
| :MEASure:STATistic:COUNt? | The query returns 200. |

### :MEASure:STATistic:RESet

- **Command Format**

  :MEASure:STATistic:RESet

- **Functional Description**

  Clear the historical statistic data and restart the statistics.

- **For Example**

| | |
|---|---|
| :MEASure:STATistic:RESet | Clear the historical statistic data and restart the statistics. |

### :MEASure:STATistic:ITEM

- **Command Format**

:MEASure:STATistic:ITEM <item>,<source1>,[<source2>]

:MEASure:STATistic:ITEM? <type>,<item>,<source1>,[<source2>]

- **Functional Description**

Open the statistics for the arbitrary waveform of the specified source and query the statistical results of the arbitrary waveform.

<item> represents waveform parameter: Refer to Measurement Parameter Table.

{VMAX|VMIN|VPP|VTOP|VBASe|VAMP|VMID|VAVG|PVAVg|VRMS|PVRMs|ACRMs|PACRms|MARea|MPARea|PMARea|NMARea|PMPArea|NMPArea|POVershoot|NOVershoot|PPReshoot|NPReshoot|PERiod|FREQuency|RTIMe|FTIMe|PWIDth|NWIDth|PDUTy|NDUTy|PPULses|NPULses|PEDGes|NEDGes|BWIDTh|BINTerval|BPERiod|BCYCles|ARRAtio|ARPRatio|STIMe|HTIMe|SHRAtio|FRFR|FRFF|FFFR|FFFF|FRLF|FRLR|FFLR|FFLF|RRPHase|FFPHase|RRDElay|FFDElay}.

<type> represents statistical type: {MAXimum|MINimum|CURRent|AVERages|DEViation}, representing maximum, minimum, current, average, and variance, respectively.

<source> represents the main source:

{CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

<source> represents the slave source:

{CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

Explanation:

If the measurement parameter is single source, set one source and omit the slave source.

If the measurement parameter requires both sources, set two sources and do not omit the slave source.

- **Return Format**

The query returns the statistical results in scientific notation.

- **For Example**

| | |
|---|---|
| :MEASure:STATistic:ITEM VAMP,CHANnel1 | Enable the amplitude measurement statistics for Channel 1. |
| :MEASure:STATistic:ITEM? MAX,VAMP,CHANnel1 | The query returns the maximum “1.120000e+00” of Channel 1. |

### :MEASure:STATistic:HISTogram:RESult?

- **Command Format**

:MEASure:STATistic:HISTogram:RESult? <item>,<source1>,[<source2>]

## Functional Description

Query the statistical results of histogram measurement. The results display the left boundary, the right boundary of the histogram, and percentage in sequence.

<item> represents waveform parameter: Refer to Measurement Parameter Table.

{VMAX|VMIN|VPP|VTOP|VBASe|VAMP|VMID|VAVG|PVAVg|VRMS|PVRMs|ACRMs|PACRms|MARea|MPARea|PMARea|NMARea|PMPArea|NMPArea|POVershoot|NOVershoot|PPReshoot|NPReshoot|PERiod|FREQuency|RTIMe|FTIMe|PWIDth|NWIDth|PDUTy|NDUTy|PPULses|NPULses|PEDGes|NEDGes|BWIDTh|BINTerval|BPERiod|BCYCles|ARRAtio|ARPRatio|STIMe|HTIMe|SHRAtio|FRFR|FRFF|FFFR|FFFF|FRLF|FRLR|FFLR|FFLF|RRPHase|FFPHase|RRDElay|FFDElay}.

<source> represents the main source:

{CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

<source> represents the slave source:

{CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

Explanation:

If the measurement parameter is single source, set one source and omit the slave source.

If the measurement parameter requires both sources, set two sources and do not omit the slave source.

## Return Format

The query returns the histogram statistical results arranged in CSV format in scientific notation. The returned data conforms to the Data Block Format.

## For Example

:MEASure:STATistic:HISTogram:RESult? VAMP,CHANnel1

The query returns the statistical results of histogram measurement for Channel 1, based on the set statistic count:

#9000000128HISTOGRAM,

Sum,Peaks,Max,Min,Pk_Pk,Mean,Median,Mode, Siqma

100, 93, -8.000mV, -16.000mV, 8.000mV, -8.400mV, -8.000mV, -8.000mV, 2.400mV

In which, #9000000148 is the header of the TMC data block, followed closely by data from the options list. The number following #9 in the data block header indicates the number of bytes of valid data following it. HISTOGRAM indicates a histogram, with each data separated by commas and each line of data separated by a newline character.

The statistical results include the following items.

Sum: Total count of statistical data.

Peaks: The maximum count of data being counted.

Max: The maximum value of the total statistical data.

Min: The minimum value of the total statistical data.

Pk_Pk: The difference between the maximum and minimum values (Max-Min) in the total statistical data.

Mean: The average of histogram.

Median: The median value of histogram.

Mode: The mode value of histogram.

Sigma: The standard deviation of histogram.

### :MEASure:THReshold:DEFault

- **Command Format**

  :MEASure:THReshold:DEFault <source>

- **Functional Description**

  Set the default threshold: the low default threshold is 10%; the middle default threshold is 50%, the high default threshold is 90%.

  <source> represents {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **For Example**

| | |
|---|---|
| :MEASure:THReshold:DEFault CHANnel1 | Set the Channel 1 threshold to the default threshold. |

### :MEASure:THReshold:MIN

- **Command Format**

  :MEASure:THReshold:MIN <source>,<value>

  :MEASure:THReshold:MIN? <source>

- **Functional Description**

  Set or query the lower limit of the threshold level when measuring the analog channel using automatic measurements.

  <source> represents {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

  <value> represents the lower limit, ranging from 5% to 93%.

- **Return Format**

  The query returns the lower limit of the threshold level in scientific notation, with the unit in %.

- **For Example**

| | |
|---|---|
| :MEASure:THReshold:MIN CHANnel1,20 | Set the lower limit of the threshold level for |

| | |
|---|---|
| | Channel 1 to 20%. |
| :MEASure:THReshold:MIN? CHANnel1 | The query returns 2.000000e+01. |

### :MEASure:THReshold:MID

- **Command Format**

  :MEASure:THReshold:MID <source>,<value>

  :MEASure:THReshold:MID? <source>

- **Functional Description**

  Set or query the middle value of the threshold level when measuring the analog channel using automatic measurements.

  <source> represents {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

  <value> represents the middle value, ranging from 6% to 94%.

- **Return Format**

  The query returns the middle value of the threshold level in scientific notation, with the unit in %.

- **For Example**

| | |
|---|---|
| :MEASure:THReshold:MID CHANnel1,30 | Set the middle value of the threshold level for Channel 1 to 30%. |
| :MEASure:THReshold:MID? CHANnel1 | The query returns 3.000000e+01. |

### :MEASure:THReshold:MAX

- **Command Format**

  :MEASure:THReshold:MAX <source>,<value>

  :MEASure:THReshold:MAX? <source>

- **Functional Description**

  Set or query the upper limit of the threshold level when measuring the analog channel using automatic measurements.

  <source> represents {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

  <value> represents the upper limit, ranging from 7% to 95%.

- **Return Format**

  The query returns the upper limit of the threshold level in scientific notation, with the unit in %.

- **For Example**

| | |
|---|---|
| :MEASure:THReshold:MAX CHANnel1,40 | Set the upper limit of the threshold level for Channel 1 to 40%. |
| :MEASure:THReshold:MAX? CHANnel1 | The query returns 4.000000e+01. |

### :MEASure:RANGe

- **Command Format**

  :MEASure:RANGe {SCReen | CURSor}

  :MEASure:RANGe?

- **Functional Description**

  Set the measurement range to SCReen (Full screen) or CURSor (Cursor area).

  When the measurement range is set to CURSor, use the commands :CURSor:MEASure and :CURSor:TYPe to enable the time cursor measurement function, and then use the commands :CURSor:CAX and :CURSor:CBX Adjust the horizontal positions of cursor A and cursor line B.

- **Return Format**

  The query returns {SCReen | CURSor}.

- **For Example**

| | |
|---|---|
| :MEASure:RANGe CURSor | Set the measurement range to CURSor (Cursor area). |
| :MEASure:RANGe? | The query returns CURSor. |

### :MEASure:AMP:TYPe

- **Command Format**

  :MEASure:AMP:TYPe {AUTO|MANual}

  :MEASure:AMP:TYPe?

- **Functional Description**

  Set or query the amplitude calculation type.

  AUTO: Automatic measurement

  MANual: Manual measurement

- **Return Format**

  The query returns {AUTO|MANual}.

- **For Example**

| | |
|---|---|
| :MEASure:AMP:TYPe AUTO | Set the amplitude calculation type to AUTO. |
| :MEASure:AMP:TYPe? | The query returns AUTO. |

### :MEASure:AMP:MANual:TOP

- **Command Format**

  :MEASure:AMP:MANual:TOP {HISTogram|MAX}

  :MEASure:AMP:MANual:TOP?

- **Functional Description**

  Set or query the manual measurement for the amplitude top value.

  HISTogram: Histogram measurement

  MAX: Maximum measurement

- **Return Format**

  The query returns {HISTogram|MAX}.

- **For Example**

| | |
|---|---|
| :MEASure:AMP:MANual:TOP MAX | Set the manual measurement for the amplitude top value to MAX. |
| :MEASure:AMP:MANual:TOP? | The query returns MAX. |

### :MEASure:AMP:MANual:BASe

- **Command Format**

  :MEASure:AMP:MANual:BASe {HISTogram|MIN}

  :MEASure:AMP:MANual:BASe?

- **Functional Description**

  Set or query the manual measurement for the amplitude bottom value.

  HISTogram: Histogram measurement; MIN: Minimum measurement.

- **Return Format**

  The query returns {HISTogram|MIN }.

- **For Example**

| | |
|---|---|
| :MEASure:AMP:MANual:BASe MIN | Set the manual measurement for the amplitude top value to MIN. |
| :MEASure:AMP:MANual:BASE? | The query returns MIN. |

### :MEASure:BURSt:TIMe

- **Command Format**

  :MEASure:BURSt:TIMe <time>

  :MEASure:BURSt:TIMe?

- **Functional Description**

Set or query the idle time of a burst during measurement.

<time>: Idle time

- **Return Format**

The query returns the idle time in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :MEASure:BURSt:TIMe 0.000002 | Set the idle time of a burst to 2 us during measurement. |
| :MEASure:BURSt:TIMe? | The query returns 2.000000e-06. |

### :MEASure:BURSt:LEVel

- **Command Format**

:MEASure:BURSt:LEVel {HIGH|LOW}

:MEASure:BURSt:LEVel?

- **Functional Description**

Set or query the idle level of a burst during measurement.

HIGH: High level

LOW: Low level

- **Return Format**

The query returns {HIGH|LOW}.

- **For Example**

| | |
|---|---|
| :MEASure:BURSt:LEVel HIGH | Set the idle level of a burst to HIGH during measurement. |
| :MEASure:BURSt:LEVel? | The query returns HIGH. |

### :MEASure:SH:CLOCk:EDGe

- **Command Format**

:MEASure:SH:CLOCk:EDGe {POSitive|NEGative|ANY}

:MEASure:SH:CLOCk:EDGe?

- **Functional Description**

Set or query the clock edge of the setup & hold trigger during measurement.

POSitive: Rising edge

NEGative: Falling edge

ANY: Arbitrary edge

- **Return Format**

The query returns {POSitive|NEGative|ANY}.

- **For Example**

| | |
|---|---|
| :MEASure:SH:CLOCk:EDGe POSitive | Set the clock edge of the setup & hold trigger to POSitive (Rising edge) during measurement. |
| :MEASure:SH:CLOCk:EDGe? | The query returns POSitive. |

### :MEASure:SH:DATA:EDGe

- **Command Format**

  :MEASure:SH:DATA:EDGe {POSitive|NEGative|ANY}

  :MEASure:SH:DATA:EDGe?

- **Functional Description**

  Set or query the data edge of the setup & hold trigger during measurement.

  POSitive: Rising edge

  NEGative: Falling edge

  ANY: Arbitrary edge

- **Return Format**

  The query returns {POSitive|NEGative|ANY}.

- **For Example**

| | |
|---|---|
| :MEASure:SH:DATA:EDGe POSitive | Set the data edge of the setup & hold trigger to POSitive (Rising edge) during measurement. |
| :MEASure:SH:DATA:EDGe? | The query returns POSitive. |

### :MEASure:RMS:UNIT

- **Command Format**

  :MEASure:RMS:UNIT {RMS|DBM|DB}

  :MEASure:RMS:UNIT?

- **Functional Description**

  Set or query the unit of RMS measurement.

- **Return Format**

  The query returns {RMS|DBM|DB}.

- **For Example**

| | |
|---|---|
| :MEASure:RMS:UNIT DB | Set the unit of RMS measurement to DB. |
| :MEASure:RMS:UNIT? | The query returns DB. |

### :MEASure:RMS:REF

- **Command Format**

:MEASure:RMS:REF <value>

:MEASure:RMS:REF?

- **Functional Description**

Set or query the reference value of RMS measurement, with the range of 0.001 to 1000.

- **Return Format**

The query returns the reference value in scientific notation.

- **For Example**

| | |
|---|---|
| :MEASure:RMS:REF 2 | Set the reference value of RMS measurement to 2. |
| :MEASure:RMS:REF? | The query returns 2.000000e+00. |

# TRIGger Command

This command is used to control the trigger sweep mode and trigger specification. The trigger determines when the oscilloscope starts sampling data and displays the waveform.

## Trigger Control

### :TRIGger:MODe

- **Command Format**

:TRIGger:MODe <mode>

:TRIGger:MODe?

- **Functional Description**

Set the trigger mode. It automatically adjusts for different models when the trigger mode is set.

<mode>: {EDGe|PULSe|VIDeo|SLOPe|RUNT|WINDow|DELay|TIMeout|DURation|SHOLd|NEDGe|PATTern|RS232|I2C|SPI|CAN|CANFD|CANXL|LIN|FR|AUDio|M1553|MANC|SENT|A429|ONEWire|I3C}

Explanation: EDGe (Edge trigger), PULSe (Pulse width trigger), VIDeo (Video trigger), SLOPe (Slope trigger), RUNT (Runt trigger), WINDow (Over-amplitude trigger), DELay (Delay trigger), TIMeout (Timeout trigger), DURation (Duration trigger), SHOLd (Setup&hold trigger), NEDGe (Nth edge trigger), PATTern (Code pattern trigger), RS232 (UART/RS232 bus trigger), I2C (I2C bus trigger), SPI (SPI bus trigger), CAN (CAN bus trigger), CANFD (CANFD bus trigger), CANXL(CANXL bus trigger), LIN (LIN bus trigger), FR (FlexRay bus trigger), AUDio (AUDIO bus trigger), M1553 (MIL - STD - 1553B bus trigger), MANC (Manchester trigger), SENT (SENT bus trigger), A429 (Arinc429 bus trigger), ONEWire (1-wire bus trigger), I3C (I3C bus trigger).

- **Return Format**

The query returns the trigger mode.

- **For Example**

| | |
|---|---|
| :TRIGger:MODe NEDGe | Set the trigger mode to NEDGe (Nth edge trigger). |
| :TRIGger:MODe? | The query returns NEDGe. |

### :TRIGger:FORCe

- **Command Format**

:TRIGger:FORCe

- **Functional Description**

Force the oscilloscope to generate a trigger signal, allowing the input waveform to be triggered and displayed even when no suitable trigger condition is found.

- **For Example**

| | |
|---|---|
| :TRIGger:FORCe | Force trigger. |

### :TRIGger:SWEep

- **Command Format**

:TRIGger:SWEep {AUTO|NORMal|SINGle}

:TRIGger:SWEep?

- **Functional Description**

Select the trigger sweep mode.

AUTO (Automatic): When no trigger condition is detected, an internal trigger signal will be generated to force a trigger.

NORMal (Normal): It can only be generated when the trigger condition is met.

SINGle (Single): Generate one trigger and stop when the trigger condition is met.

- **Return Format**

The query returns the trigger sweep mode {AUTO|NORMal|SINGle}.

- **For Example**

| | |
|---|---|
| :TRIGger:SWEep AUTO | Set the trigger sweep mode of Channel 1 to AUTO. |
| :TRIGger:SWEep? | The query returns AUTO. |

### :TRIGger:COUPling

- **Command Format**

:TRIGger:COUPling {DC|AC|LF|HF}

:TRIGger:COUPling?

- **Functional Description**

  Set the coupling mode.

  DC: Allows both DC and AC components to pass.

  AC: Blocks all components.

  LF: Blocks the DC component and rejects low-frequency components.

  HF: Rejects high-frequency components.

- **Return Format**

  The query returns the coupling mode {DC|AC|LF|HF}.

- **For Example**

| | |
|---|---|
| :TRIGger:COUPling AC | Set the coupling mode of edge trigger to AC. |
| :TRIGger:COUPling? | The query returns AC. |

### :TRIGger:HOLDoff

- **Command Format**

  :TRIGger:HOLDoff <time>

  :TRIGger:HOLDoff?

- **Functional Description**

  Set the trigger holdoff time, ranging from 0 to 10 s.

- **Return Format**

  The query returns the trigger holdoff time in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:HOLDoff 1s | Set the trigger holdoff time to 1s. |
| :TRIGger:HOLDoff? | The query returns 1.000000e+00. |

### :TRIGger:NREJect

- **Command Format**

  :TRIGger:NREJect { {1|ON} | {0|OFF} }

  :TRIGger:NREJect?

- **Functional Description**

  Set and query the reject state of noise trigger to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :TRIGger:NREJect ON | Enable noise trigger reject. |

| | |
|---|---|
| :TRIGger:NREJect? | The query returns 1, indicating that noise trigger reject is enabled. |

### :TRIGger:SENSitivity:ENABle

- **Command Format**

  :TRIGger:SENSitivity:ENABle { {1|ON} | {0|OFF} }

  :TRIGger:SENSitivity:ENABle?

- **Functional Description**

  Set and query the state of trigger sensitivity.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :TRIGger:SENSitivity:ENABle ON | Enable trigger sensitivity. |
| :TRIGger:SENSitivity:ENABle? | The query returns 1. |

### :TRIGger:SENSitivity

- **Command Format**

  :TRIGger:SENSitivity <sensitivity>

  :TRIGger:SENSitivity?

- **Functional Description**

  Set and query the trigger sensitivity range, with the range from 0 to 100.

- **Return Format**

  The query returns the trigger sensitivity range in scientific notation, with the unit in %.

- **For Example**

| | |
|---|---|
| :TRIGger:SENSitivity 10 | Set the trigger sensitivity range to 10%. |

### :TRIGger:STATus?

- **Command Format**

  :TRIGger:STATus?

- **Functional Description**

  Query the running state of the current trigger.

- **Return Format**

  The query returns STOP/ARMED/READY/TRIGED/AUTO/SCAN/RESET/REPLAY/WAIT.

- **For Example**

| | |
|---|---|
| :TRIGger:STATus? | The query returns AUTO. |

## Zone Triggering

### :TRIGger:AREa

- **Command Format**

  :TRIGger:AREa { {1|ON} | {0|OFF} }

  :TRIGger:AREa?

- **Functional Description**

  Set or query the zone trigger state: ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :TRIGger:AREa ON Enable the zone trigger state.

  :TRIGger:AREa? The query returns 1, indicating that the zone trigger state is enabled.

### :TRIGger:AREa:A

- **Command Format**

  :TRIGger:AREa:A <hp1>,<vp1>,<hp2>,<vp2>

  :TRIGger:AREa:A?

- **Functional Description**

  Set or query Zone A, use the coordinates of the upper left and bottom right corners to define the screen range. The left boundary should be less than the right boundary, and the upper boundary should be greater than the lower boundary.

  <hp1>: Represents the horizontal time value of the upper left point of the area, with the unit in seconds (s).

  <vp1>: Represents the channel vertical value of the upper left point of the area. The unit is determined by the channel's vertical unit.

  <hp2>: Represents the horizontal time value of the bottom right point of the area, with the unit in seconds (s).

  <vp2>: Represents the channel vertical value of the bottom right point of the area. The unit is determined by the channel's vertical unit.

- **Return Format**

  The query returns the coordinate value in scientific notation.

- **For Example**

  :TRIGger:AREa:A -5us,200mv,5us,-200mv

  Zone A is from the upper left point [-5 us, 200 mV] to the bottom right point [5 us, -200 mV].

:TRIGger:AREa:A?

The query returns -5.000000e-06, 2.000000e-01, 5.000000e-06, -2.000000e-01.

### :TRIGger:AREa:A:ENABle

- **Command Format**

:TRIGger:AREa:A:ENABle { {1|ON} | {0|OFF} }

:TRIGger:AREa:A:ENABle?

- **Functional Description**

Set or query the enabling state of Zone A.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :TRIGger:AREa:A:ENABle ON | Enable Zone A trigger. |
| :TRIGger:AREa:A:ENABle? | The query returns 1, indicating that Zone A trigger is enabled. |

### :TRIGger:AREa:A:SOURce

- **Command Format**

:TRIGger:AREa:A:SOURce <source>

:TRIGger:AREa:A:SOURce?

- **Functional Description**

Set or query the trigger source of Zone A.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4 }.

- **Return Format**

The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :TRIGger:AREa:A:SOURce CHANnel1 | Set the trigger source of Zone A to Channel 1. |
| :TRIGger:AREa:A:SOURce? | The query returns CHANnel1. |

### :TRIGger:AREa:A:INTersect

- **Command Format**

:TRIGger:AREa:A:INTersect { {1|ON} | {0|OFF} }

:TRIGger:AREa:A:INTersect?

- **Functional Description**

Set or query the trigger intersection state of Zone A.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :TRIGger:AREa:A:INTersect ON Set Zone A trigger to intersect.

  :TRIGger:AREa:A:INTersect? The query returns 1, indicating Zone A trigger is intersected.

### :TRIGger:AREa:B

- **Command Format**

  :TRIGger:AREa:B <hp1>,<vp1>,<hp2>,<vp2>

  :TRIGger:AREa:B?

- **Functional Description**

  Set or query Zone B, use the coordinates of the upper left and bottom right corners to define the screen range. The left boundary should be less than the right boundary, and the upper boundary should be greater than the lower boundary.

  <hp1>: Represents the horizontal time value of the upper left point of the area, with the unit in seconds (s).

  <vp1>: Represents the channel vertical value of the upper left point of the area. The unit is determined by the channel's vertical unit.

  <hp2>: Represents the horizontal time value of the bottom right point of the area, with the unit in seconds (s).

  <vp2>: Represents the channel vertical value of the bottom right point of the area. The unit is determined by the channel's vertical unit.

- **Return Format**

  The query returns the coordinate value in scientific notation.

- **For Example**

  :TRIGger:AREa:B -5us,200mv,5us,-200mv

  Zone B is from the upper left point [-5 us, 200 mV] to the bottom right point [5 us, -200 mV].

  :TRIGger:AREa:B?

  The query returns -5.000000e-06, 2.000000e-01, 5.000000e-06, and -2.000000e-01.

### :TRIGger:AREa:B:ENABle

- **Command Format**

  :TRIGger:AREa:B:ENABle { {1|ON} | {0|OFF} }

  :TRIGger:AREa:B:ENABle?

- **Functional Description**

  Set or query the enabling state of Zone B.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :TRIGger:AREa:B:ENABle ON Enable Zone B trigger.

  :TRIGger:AREa:B:ENABle? The query returns 1, indicating that Zone B trigger is enabled.

### :TRIGger:AREa:B:SOURce

- **Command Format**

  :TRIGger:AREa:B:SOURce <source>

  :TRIGger:AREa:B:SOURce?

- **Functional Description**

  Set or query the trigger source of Zone B.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4 }.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

  :TRIGger:AREa:B:SOURce CHANnel1 Set the trigger source of Zone B to Channel 1.

  :TRIGger:AREa:B:SOURce? The query returns CHANnel1.

### :TRIGger:AREa:B:INTersect

- **Command Format**

  :TRIGger:AREa:B:INTersect { {1|ON} | {0|OFF} }

  :TRIGger:AREa:B:INTersect?

- **Functional Description**

  Set or query the trigger intersection state of Zone B.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :TRIGger:AREa:B:INTersect ON Set Zone B trigger to intersect.

  :TRIGger:AREa:B:INTersect? The query returns 1, indicating Zone B trigger is intersected.

## Edge Triggering

### :TRIGger:EDGe:SOURce

- **Command Format**

  :TRIGger:EDGe:SOURce <source>

  :TRIGger:EDGe:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4<Dx>}.

  CHANnel<n>: Physical channel

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:EDGe:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:EDGe:SOURce? | The query returns CHANnel1. |

### :TRIGger:EDGe:LEVel

- **Command Format**

  :TRIGger:EDGe:LEVel <level>

  :TRIGger:EDGe:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:EDGe:LEVel 2 | Set the trigger level to 2 V. |
| :TRIGger:EDGe:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:EDGe:POLarity

- **Command Format**

  :TRIGger:EDGe:POLarity {POSitive|NEGative|ANY}

  :TRIGger:EDGe:POLarity?

- **Functional Description**

  Set the trigger edge type to POSitive (Rising edge), NEGative (Falling edge), or ANY (Arbitrary edge).

- **Return Format**

  The query returns the trigger edge type { POSitive | NEGative | ANY }.

- **For Example**

| | |
|---|---|
| :TRIGger:EDGe:POLarity POS | Set the trigger edge type to POSitive (Rising edge). |
| :TRIGger:EDGe:POLarity? | The query returns POSitive. |

## Pulse Width Triggering

### :TRIGger:PULSe:SOURce

- **Command Format**

  :TRIGger:PULse:SOURce <source>

  :TRIGger:PULse:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

  CHANnel<n>: Physical channel

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:PULse:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:PULse:SOURce? | The query returns CHANnel1. |

### :TRIGger:PULSe:LEVel

- **Command Format**

  :TRIGger:PULse:LEVel <level>

  :TRIGger:PULse:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current

amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:PULse:LEVel 2 | Set the trigger level to 2 V. |
| :TRIGger:PULse:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:PULSe:QUALifier

- **Command Format**

:TRIGger:PULSe:QUALifier {GREaterthan | LESSthan | INRange}

:TRIGger:PULSe:QUALifier?

- **Functional Description**

Set the trigger condition of pulse time to GREaterthan (Greater than), LESSthan (Less than), or INRange (Within the range).

- **Return Format**

The query returns {GREaterthan | LESSthan | INRange}.

- **For Example**

| | |
|---|---|
| :TRIGger:PULSe:QUALifier GRE | Set the pulse condition to GREaterthan. |
| :TRIGger:PULSe:QUALifier? | The query returns GREaterthan. |

### :TRIGger:PULSe:POLarity

- **Command Format**

:TRIGger:PULSe:POLarity {POSitive | NEGative}

:TRIGger:PULSe:POLarity?

- **Functional Description**

Set the pulse polarity to POSitive (Positive pulse width) or NEGative (Negative pulse width).

- **Return Format**

The query returns { POSitive | NEGative }.

- **For Example**

| | |
|---|---|
| :TRIGger:PULSe:POL POS | Set the pulse polarity to POSitive (Positive pulse width). |
| :TRIGger:PULSe:POL? | The query returns POSitive. |

### :TRIGger:PULSe:TIMe:UPPer

- **Command Format**

:TRIGger:PULSe:TIMe:UPPer <time>

:TRIGger:PULSe:TIMe:UPPer?

- **Functional Description**

  Set the upper limit time for the pulse width trigger.

- **Return Format**

  The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:PULSe:TIMe:UPPer 1 | Set the upper limit time for the pulse width trigger to 1s. |
| :TRIGger:PULSe:TIMe:UPPer? | The query returns 1.000000e+00. |

### :TRIGger:PULSe:TIMe:LOWer

- **Command Format**

  :TRIGger:PULSe:TIMe:LOWer <time>

  :TRIGger:PULSe:TIMe:LOWer?

- **Functional Description**

  Set the lower limit time for the pulse width trigger.

- **Return Format**

  The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:PULSe:TIMe:LOWer 1 | Set the lower limit time for the pulse width trigger to 1s. |
| :TRIGger:PULSe:TIMe:LOWer? | The query returns 1.000000e+00. |

## Video Triggering

### :TRIGger:VIDeo:SOURce

- **Command Format**

  :TRIGger:VIDeo:SOURce <source>

  :TRIGger:VIDeo:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :TRIGger:VIDeo:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:VIDeo:SOURce? | The query returns CHANnel1. |

### :TRIGger:VIDeo:LEVel

- **Command Format**

  :TRIGger:VIDeo:LEVel <level>

  :TRIGger:VIDeo:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:VIDeo:LEVel 2 | Set the trigger level to 2 V. |
| :TRIGger:VIDeo:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:VIDeo:MODe

- **Command Format**

  :TRIGger:VIDeo:MODe {ODD|EVEN|LINe|ALINes|ALL}

  :TRIGger:VIDeo:MODe?

- **Functional Description**

  Set the sync mode of the video trigger to ODD, EVEN, LINe (Specified line), ALINes (All lines), or ALL.

- **Return Format**

  The query returns {ODD|EVEN|LINe|ALINes|ALL}.

- **For Example**

| | |
|---|---|
| :TRIGger:VIDeo:MODe ODD | Set the sync mode of the video trigger to ODD. |
| :TRIGger:VIDeo:MODe? | The query returns ODD. |

### :TRIGger:VIDeo:STANdard

- **Command Format**

  :TRIGger:VIDeo:STANdard <standard>

  :TRIGger:VIDeo:STANdard?

- **Functional Description**

  Set the video standard.

  <standard>:

{NTSC|PAL|SECAM|R525P60|R625P50|R720P24|R720P25|R720P30|R720P50|R720P60|R1080I25|R1080I30|R1080P24|R1080P25|R1080P30|R1080PSF24}

- **Return Format**

  The query returns {NTSC|PAL|SECAM|R525P60|R625P50|R720P24|R720P25|R720P30|R720P50|R720P60|R1080I25|R1080I30|R1080P24|R1080P25|R1080P30|R1080PSF24}.

- **For Example**

| | |
|---|---|
| :TRIGger:VIDeo:STANdard NTSC | Set the video standard to NTSC. |
| :TRIGger:VIDeo:STANdard? | The query returns NTSC. |

### :TRIGger:VIDEO:LINe

- **Command Format**

  :TRIGger:VIDEO:LINe <value>

  :TRIGger:VIDEO:LINe?

- **Functional Description**

  Set the specified line for video sync, where <value> represents the specified line within the range determined by the video standard.

- **Return Format**

  The query returns the currently specified lines.

- **For Example**

| | |
|---|---|
| :TRIG:VIDEO:LINe 50 | Set the specified line for video sync to 50. |
| :TRIG:VIDEO:LINe? | The query returns 50. |

## Slope Triggering

### :TRIGger:SLOPe:SOURce

- **Command Format**

  :TRIGger:SLOPe:SOURce <source>

  :TRIGger:SLOPe:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :TRIGger:SLOPe:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:SLOPe:SOURce? | The query returns CHANnel1. |

### :TRIGger:SLOPe:LOW:LEVel

- **Command Format**

:TRIGger:SLOPe:LOW:LEVel <level>

:TRIGger:SLOPe:LOW:LEVel?

- **Functional Description**

Set or query the low trigger level value.

<level>: Trigger level value

- **Return Format**

The query returns the low trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:SLOPe:LOW:LEVel -3 | Set the low trigger level value to -3 V. |
| :TRIGger:SLOPe:LOW:LEVel? | The query returns -3.000000e+00. |

### :TRIGger:SLOPe:HIGH:LEVel

- **Command Format**

:TRIGger:SLOPe:HIGH:LEVel <level>

:TRIGger:SLOPe:HIGH:LEVel?

- **Functional Description**

Set or query the high trigger level value.

<level>: Trigger level value

- **Return Format**

The query returns the high trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:SLOPe:HIGH:LEVel 3 | Set the high trigger level value to 3 V. |
| :TRIGger:SLOPe:HIGH:LEVel? | The query returns 3.000000e+00. |

### :TRIGger:SLOPe:QUALifier

- **Command Format**

:TRIGger:SLOPe:QUALifier {GREaterthan | LESSthan | INRange}

:TRIGger:SLOPe:QUALifier?

- **Functional Description**

Set the trigger condition for slope time to GREaterthan (Greater than), LESSthan (Less than), or INRange (Within the range).

- **Return Format**

The query returns {GREaterthan | LESSthan | INRange}.

- **For Example**

| | |
|---|---|
| :TRIGger:SLOPe:QUALifier GRE | Set the slope condition to GREaterthan. |
| :TRIGger:SLOPe:QUALifier? | The query returns GREaterthan. |

### :TRIGger:SLOPe:POLarity

- **Command Format**

:TRIGger:SLOPe:POLarity {POSitive|NEGative}

:TRIGger:SLOPe:POLarity?

- **Functional Description**

Set the slope trigger type to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns {POSitive|NEGative}.

- **For Example**

| | |
|---|---|
| :TRIGger:SLOPe:POLarity POS | Set the slope trigger type to POSitive (Rising edge). |
| :TRIGger:SLOPe:POLarity? | The query returns POSitive. |

### :TRIGger:SLOPe:TIMe:UPPer

- **Command Format**

:TRIGger:SLOPe:TIMe:UPPer <time>

:TRIGger:SLOPe:TIMe:UPPer?

- **Functional Description**

Set the upper limit time for the slope trigger.

- **Return Format**

The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:SLOPe:TIMe:UPPer 1 | Set the upper limit time for the slope trigger to 1s. |
| :TRIGger:SLOPe:TIMe:UPPer? | The query returns 1.000000e+00. |

### :TRIGger:SLOPe:TIMe:LOWer

- **Command Format**

  :TRIGger:SLOPe:TIMe:LOWer <time>

  :TRIGger:SLOPe:TIMe:LOWer?

- **Functional Description**

  Set the lower limit time for the slope trigger.

- **Return Format**

  The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:SLOPe:TIMe:LOWer 1 | Set the lower limit time for the slope trigger to 1s. |
| :TRIGger:SLOPe:TIMe:LOWer? | The query returns 1.000000e+00. |

## Runt Triggering

### :TRIGger:RUNT:SOURce

- **Command Format**

  :TRIGger:RUNT:SOURce <source>

  :TRIGger:RUNT:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :TRIGger:RUNT:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:RUNT:SOURce? | The query returns CHANnel1. |

### :TRIGger:RUNT:LOW:LEVel

- **Command Format**

  :TRIGger:RUNT:LOW:LEVel <level>

  :TRIGger:RUNT:LOW:LEVel?

- **Functional Description**

  Set or query the low trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the low-trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

  :TRIGger:RUNT:LOW:LEVel -3 Set the low trigger level value to -3 V.

  :TRIGger:RUNT:LOW:LEVel? The query returns -3.000000e+00.

### :TRIGger:RUNT:HIGH:LEVel

- **Command Format**

  :TRIGger:RUNT:HIGH:LEVel <level>

  :TRIGger:RUNT:HIGH:LEVel?

- **Functional Description**

  Set or query the high trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the high-trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

  :TRIGger:RUNT:HIGH:LEVel 3 Set the high trigger level value to 3 V.

  :TRIGger:RUNT:HIGH:LEVel? The query returns 3.000000e+00.

### :TRIGger:RUNT:QUALifier

- **Command Format**

  :TRIGger:RUNT:QUALifier {GREaterthan | LESSthan | INRange | NONE}

  :TRIGger:RUNT:QUALifier?

- **Functional Description**

  Set the time search for runt trigger to GREaterthan (Greater than), LESSthan (Less than), INRange (Within the range), or NONE (Arbitrary).

- **Return Format**

  The query returns {GREaterthan | LESSthan | INRange | NONE}.

- **For Example**

  :TRIGger:RUNT:QUALifier GRE Set the slope condition to GREaterthan.

  :TRIGger:RUNT:QUALifier? The query returns GREaterthan.

### :TRIGger:RUNT:POLarity

- **Command Format**

  :TRIGger:RUNT:POLarity {POSitive | NEGative}

  :TRIGger:RUNT:POLarity?

- **Functional Description**

  Set the runt polarity to POSitive (Positive pulse width) or NEGative (Negative pulse width).

- **Return Format**

  The query returns {POSitive | NEGative}.

- **For Example**

  :TRIGger:RUNT:POL POS Set the pulse polarity to POSitive (Positive pulse width).

  :TRIGger:RUNT:POL? The query returns POSitive.

### :TRIGger:RUNT:TIMe:UPPer

- **Command Format**

  :TRIGger:RUNT:TIMe:UPPer <time>

  :TRIGger:RUNT:TIMe:UPPer?

- **Functional Description**

  Set the upper limit time for the runt trigger.

- **Return Format**

  The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

  :TRIGger:RUNT:TIMe:UPPer 1 Set the upper limit time for the runt trigger to 1s.

  :TRIGger:RUNT:TIMe:UPPer? The query returns 1.000000e+00.

### :TRIGger:RUNT:TIMe:LOWer

- **Command Format**

  :TRIGger:RUNT:TIMe:LOWer <time>

  :TRIGger:RUNT:TIMe:LOWer?

- **Functional Description**

  Set the lower limit time for the runt trigger.

- **Return Format**

  The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

  :TRIGger:RUNT:TIMe:LOWer 1 Set the lower limit time for the runt trigger to 1s.

| | |
|---|---|
| :TRIGger:RUNT:TIMe:LOWer? | The query returns 1.000000e+00. |

## Over-amplitude Triggering

### :TRIGger:WINDow:SOURce

- **Command Format**

  :TRIGger:WINDow:SOURce <source>

  :TRIGger:WINDow:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :TRIGger:WINDow:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:WINDow:SOURce? | The query returns CHANnel1. |

### :TRIGger:WINDow:LOW:LEVel

- **Command Format**

  :TRIGger:WINDow:LOW:LEVel <level>

  :TRIGger:WINDow:LOW:LEVel?

- **Functional Description**

  Set or query the low trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the low-trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:WINDow:LOW:LEVel -3 | Set the low trigger level value to -3 V. |
| :TRIGger:WINDow:LOW:LEVel? | The query returns -3.000000e+00. |

### :TRIGger:WINDow:HIGH:LEVel

- **Command Format**

  :TRIGger:WINDow:HIGH:LEVel <level>

:TRIGger:WINDow:HIGH:LEVel?

- **Functional Description**

Set or query the high trigger level value.

<level>: Trigger level value

- **Return Format**

The query returns the high-trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:WINDow:HIGH:LEVel 3 | Set the high trigger level value to 3 V. |
| :TRIGger:WINDow:HIGH:LEVel? | The query returns 3.000000e+00. |

### :TRIGger:WINDow:POLarity

- **Command Format**

:TRIGger:WINDow:POLarity {POSitive|NEGative|ANY}

:TRIGger:WINDow:POLarity?

- **Functional Description**

Set the trigger edge type to POSitive (Rising edge), NEGative (Falling edge), or ANY (Arbitrary edge).

- **Return Format**

The query returns the trigger edge type {POSitive|NEGative|ANY}.

- **For Example**

| | |
|---|---|
| :TRIGger:WINDow:POLarity POS | Set the window trigger to POSitive (Rising edge). |
| :TRIGger:WINDow:POLarity? | The query returns POSitive. |

### :TRIGger:WINDow:TIMe

- **Command Format**

:TRIGger:WINDow:TIMe <time>

:TRIGger:WINDow:TIMe?

- **Functional Description**

Set the time interval for the window trigger.

- **Return Format**

The query returns the current time interval, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:WINDow:TIMe 1 | Set the time interval for the window trigger to 1s. |

| | |
|---|---|
| :TRIGger:WINDow:TIMe? | The query returns 1.000000e+00. |

### :TRIGger:WINDow:POSition

- **Command Format**

  :TRIGger:WINDow:POSition {ENTer|EXIT|TIMe}

  :TRIGger:WINDow:POSition?

- **Functional Description**

  Set the window trigger position.

- **Return Format**

  The query returns {ENTer|EXIT|TIMe}.

- **For Example**

| | |
|---|---|
| :TRIGger:WINDow:POS TIMe | Set the window trigger position to TIMe. |
| :TRIGger:WINDow:POS? | The query returns TIMe. |

## Delay Triggering

### :TRIGger:DELay:ASOURce

- **Command Format**

  :TRIGger:DELay:ASOURce {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}

  :TRIGger:DELay:ASOURce?

- **Functional Description**

  Set the source 1 for the delay trigger.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:ASOURce CHAN1 | Set the trigger source 1 as Channel 1. |
| :TRIGger:DELay:ASOURce? | The query returns CHANnel1. |

### :TRIGger:DELay:ALEVel

- **Command Format**

  :TRIGger:DELay:ALEVel <level>

  :TRIGger:DELay:ALEVel?

- **Functional Description**

  Set or query the trigger level of source 1.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:ALEVel 2 | Set the trigger level of source 1 to 2 V. |
| :TRIGger:DELay:ALEVel? | The query returns 2.000000e+00. |

### :TRIGger:DELay:APOLarity

- **Command Format**

:TRIGger:DELay:APOLarity {NEGative | POSitive}

:TRIGger:DELay:APOLarity?

- **Functional Description**

Set the edge type for trigger source 1 to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns {NEGative | POSitive}.

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:APOLarity NEG | Set the edge type for trigger source 1 to NEGative. |
| :TRIGger:DELay:APOLarity? | The query returns NEGative. |

### :TRIGger:DELay:BSOURce

- **Command Format**

:TRIGger:DELay:BSOURce {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}

:TRIGger:DELay:BSOURce?

- **Functional Description**

Set the source 2 for the delay trigger.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:BSOURce CHAN1 | Set the trigger source 2 as Channel 1. |
| :TRIGger:DELay:BSOURce? | The query returns CHANnel1. |

### :TRIGger:DELay:BLEVel

- **Command Format**

  :TRIGger:DELay:BLEVel <level>

  :TRIGger:DELay:BLEVel?

- **Functional Description**

  Set or query the trigger level of source 2.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:BLEVel 2 | Set the trigger level of source 2 to 2 V. |
| :TRIGger:DELay:BLEVel? | The query returns 2.000000e+00. |

### :TRIGger:DELay:BPOLarity

- **Command Format**

  :TRIGger:DELay:BPOLarity {NEGative | POSitive}

  :TRIGger:DELay:BPOLarity?

- **Functional Description**

  Set the edge type for trigger source 2 to POSitive (Rising) or NEGative (Falling).

- **Return Format**

  The query returns {NEGative | POSitive}.

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:BPOLarity NEG | Set the edge type for trigger source 2 to NEGative. |
| :TRIGger:DELay:BPOLarity? | The query returns NEGative. |

### :TRIGger:DELay:QUALifier

- **Command Format**

  :TRIGger:DELay:QUALifier { GREaterthan | LESSthan | INRange | OUTRange }

  :TRIGger:DELay:QUALifier?

- **Functional Description**

  Set the time interval for the delay trigger to GREaterthan (Greater than), LESSthan (Less than), INRange (Within the range), or OUTRange (Outside of the range).

- **Return Format**

The query returns { GREaterthan | LESSthan | INRange | OUTRange }.

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:QUALifier GRE | Set the slope condition to GREaterthan. |
| :TRIGger:DELay:QUALifier? | The query returns GREaterthan. |

### :TRIGger:DELay:TIMe:UPPer

- **Command Format**

:TRIGger:DELay:TIMe:UPPer <time>

:TRIGger:DELay:TIMe:UPPer?

- **Functional Description**

Set the upper limit time for the delay trigger.

- **Return Format**

The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:TIMe:UPPer 1 | Set the upper limit time for the delay trigger to 1s. |
| :TRIGger:DELay:TIMe:UPPer? | The query returns 1.000000e+00. |

### :TRIGger:DELay:TIMe:LOWer

- **Command Format**

:TRIGger:DELay:TIMe:LOWer <time>

:TRIGger:DELay:TIMe:LOWer?

- **Functional Description**

Set the lower limit time for the delay trigger.

- **Return Format**

The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:DELay:TIMe:LOWer 1 | Set the lower limit time for the delay trigger to 1s. |
| :TRIGger:DELay:TIMe:LOWer? | The query returns 1.000000e+00. |

## Timeout Triggering

### :TRIGger:TIMeout:SOURce

- **Command Format**

:TRIGger:TIMeout:SOURce <source>

:TRIGger:TIMeout:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:TIMeout:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:TIMeout:SOURce? | The query returns CHANnel1. |

### :TRIGger:TIMeout:LEVel

- **Command Format**

  :TRIGger:TIMeout:LEVel <level>

  :TRIGger:TIMeout:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:TIMeout:LEVel -3 | Set the trigger level to -3 V. |
| :TRIGger:TIMeout:LEVel? | The query returns -3.000000e+00. |

### :TRIGger:TIMeout:TIMe

- **Command Format**

  :TRIGger:TIMeout:TIMe <time>

  :TRIGger:TIMeout:TIMe?

- **Functional Description**

  Set the time interval for the timeout trigger.

- **Return Format**

  The query returns the current time interval, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:TIMeout:TIMe 1 | Set the time interval for the timeout trigger to 1s. |
| :TRIGger:TIMeout:TIMe? | The query returns 1.000000e+00. |

### :TRIGger:TIMeout:POLarity

- **Command Format**

  :TRIGger:TIMeout:POLarity {POSitive|NEGative|ANY}

  :TRIGger:TIMeout:POLarity?

- **Functional Description**

  Set the trigger edge type to POSitive (Rising edge), NEGative (Falling edge), or ANY (Arbitrary edge).

- **Return Format**

  The query returns the trigger edge type {POSitive|NEGative|ANY}.

- **For Example**

| | |
|---|---|
| :TRIGger:TIMeout:POLarity POS | Set the trigger edge type to POSitive (Rising edge). |
| :TRIGger:TIMeout:POLarity? | The query returns POSitive. |

## Duration Triggering

### :TRIGger:DURation:LEVel

- **Command Format**

  :TRIGger:DURation:LEVel <level>

  :TRIGger:DURation:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:DURation:LEVel 3 | Set the trigger level to 3 V. |
| :TRIGger:DURation:LEVel? | The query returns 3.000000e+00. |

### :TRIGger:DURation:PATTern

- **Command Format**

  :TRIGger:DURation:PATTern <source>,<pch>

:TRIGger:DURation:PATTern? <source>

- **Functional Description**

Set or query the trigger code pattern for the specified source. X represents the default value.

<source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

<pch>: {H|L|X}.

- **Return Format**

The query returns the current trigger code pattern of the specified source.

- **For Example**

| | |
|---|---|
| :TRIGger:DURation:PATTern CHANnel1,H | Set the code pattern of Channel 1 as H. |
| :TRIGger:DURation:PATTern? CHANnel1 | The query returns H. |

### :TRIGger:DURation:QUALifier

- **Command Format**

:TRIGger:DURation:QUALifier { GREaterthan | LESSthan | INRange }

:TRIGger:DURation:QUALifier?

- **Functional Description**

Set the time interval for trigger delay time to GREaterthan (Greater than), LESSthan (Less than), or INRange (Within the range).

- **Return Format**

The query returns { GREaterthan | LESSthan | INRange }.

- **For Example**

| | |
|---|---|
| :TRIGger:DURation:QUALifier GRE | Set the slope condition to GREaterthan. |
| :TRIGger:DURation:QUALifier? | The query returns GREaterthan. |

### :TRIGger:DURation:TIMe:LOWer

- **Command Format**

:TRIGger:DURation:TIMe:LOWer <time>

:TRIGger:DURation:TIMe:LOWer?

- **Functional Description**

Set the lower limit time for the duration trigger. The lower limit time can be set when the time interval is “GREaterthan.”

- **Return Format**

The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

:TRIGger:DURation:TIMe:LOWer 1 Set the lower limit time for the duration trigger to 1s.

:TRIGger:DURation:TIMe:LOWer? The query returns 1.000000e+00.

### :TRIGger:DURation:TIMe:UPPer

- **Command Format**

:TRIGger:DURation:TIMe:UPPer <time>

:TRIGger:DURation:TIMe:UPPer?

- **Functional Description**

Set the upper limit time for the duration trigger. The upper limit time can be set when the time interval is LESSthan.

- **Return Format**

The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

:TRIGger:DURation:TIMe:UPPer 1 Set the upper limit time for the duration trigger to 1s.

:TRIGger:DURation:TIMe:UPPer? The query returns 1.000000e+00.

## Setup & Hold Trigger

### :TRIGger:SHOLd:SDA

- **Command Format**

:TRIGger:SHOLd:SDA {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}

:TRIGger:SHOLd:SDA?

- **Functional Description**

Set the data source for the setup & hold trigger.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

:TRIGger:SHOLd:SDA CHAN1 Set Channel 1 as data source.

:TRIGger:SHOLd:SDA? The query returns CHANnel1.

### :TRIGger:SHOLd:DLEVel

- **Command Format**

:TRIGger:SHOLd:DLEVel<level>

:TRIGger:SHOLd:DLEVel?

- **Functional Description**

Set or query the trigger level value.

<level>: Trigger level value

- **Return Format**

The query returns the trigger data level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:SHOLd:DLEVel 3 | Set the trigger level value to 3 V. |
| :TRIGger:SHOLd:DLEVel? | The query returns 3.000000e+00. |

### :TRIGger:SHOLd:SCL

- **Command Format**

:TRIGger:SHOLd:SCL {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}

:TRIGger:SHOLd:SCL?

- **Functional Description**

Set the data source for the setup & hold trigger.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:SHOLd:SCL CHAN1 | Set Channel 1 as clock source. |
| :TRIGger:SHOLd:SCL? | The query returns CHANnel1. |

### :TRIGger:SHOLd:CLEVel

- **Command Format**

:TRIGger:SHOLd:CLEVel<level>

:TRIGger:SHOLd:CLEVel?

- **Functional Description**

Set or query trigger clock level value.

<level>: Trigger level value

- **Return Format**

The query returns the trigger clock level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:SHOLd:CLEVel 3 | Set the trigger clock level value to 3 V. |
| :TRIGger:SHOLd:CLEVel? | The query returns 3.000000e+00. |

### :TRIGger:SHOLd:POLarity

- **Command Format**

:TRIGger:SHOLd:POLarity {POSitive|NEGative}

:TRIGger:SHOLd:POLarity?

- **Functional Description**

Set the edge type for the setup & hold trigger to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns {POSitive|NEGative}.

- **For Example**

| | |
|---|---|
| :TRIGger:SHOLd:POLarity POS | Set the edge type for the setup & hold trigger to POSitive (Rising edge). |
| :TRIGger:SHOLd:POLarity? | The query returns POSitive. |

### :TRIGger:SHOLd:PATTern

- **Command Format**

:TRIGger:SHOLd:PATTern { HIGH | LOW }

:TRIGger:SHOLd:PATTern?

- **Functional Description**

Set or query the data type for the setup & hold trigger to HIGH (High level) or LOW (Low level).

- **Return Format**

The query returns { HIGH | LOW }.

- **For Example**

| | |
|---|---|
| :TRIGger:SHOLd:PATTern HIGH | Set the data type for setup & hold trigger to HIGH (High level). |
| :TRIGger:SHOLd:PATTern? | The query returns HIGH. |

### :TRIGger:SHOLd:QUALifier

- **Command Format**

:TRIGger:SHOLd:QUALifier { SETup | HOLD | SH }

:TRIGger:SHOLd:QUALifier?

- **Functional Description**

  Set the trigger condition to SETup (Setup time), HOLD (Hold time), or SH (Setup and Hold time).

- **Return Format**

  The query returns { SETup | HOLD | SH }.

- **For Example**

| | |
|---|---|
| :TRIGger:SHOLd:QUALifier HOLD | Set the trigger condition to HOLD. |
| :TRIGger:SHOLd:QUALifier? | The query returns HOLD. |

### :TRIGger:SHOLd:TIMe

- **Command Format**

  :TRIGger:SHOLd:TIMe <time>

  :TRIGger:SHOLd:TIMe?

- **Functional Description**

  Set the time interval for the setup & hold trigger.

- **Return Format**

  The query returns the current time interval, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:SHOLd:TIMe 1 | Set the time interval for the setup & hold trigger to 1s. |
| :TRIGger:SHOLd:TIMe? | The query returns 1.000000e+00. |

## Nth Edge Triggering

### :TRIGger:NEDGe:SOURce

- **Command Format**

  :TRIGger:NEDGe:SOURce <source>

  :TRIGger:NEDGe:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :TRIGger:NEDGe:SOURce CHANnel1 | Set the trigger source as Channel 1. |

| | |
|---|---|
| :TRIGger:NEDGe:SOURce? | The query returns CHANnel1. |

### :TRIGger:NEDGe:LEVel

- **Command Format**

  :TRIGger:NEDGe:LEVel <level>

  :TRIGger:NEDGe:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:NEDGe:LEVel -3 | Set the trigger level to -3 V. |
| :TRIGger:NEDGe:LEVel? | The query returns -3.000000e+00. |

### :TRIGger:NEDGe:POLarity

- **Command Format**

  :TRIGger:NEDGe:POLarity {POSitive|NEGative}

  :TRIGger:NEDGe:POLarity?

- **Functional Description**

  Set the trigger edge type to POSitive (Rising) or NEGative (Falling).

- **Return Format**

  The query returns the trigger edge type {POSitive|NEGative }.

- **For Example**

| | |
|---|---|
| :TRIGger:NEDGe:POLarity POS | Set the trigger edge type to POSitive (Rising edge). |
| :TRIGger:NEDGe:POLarity? | The query returns POSitive. |

### :TRIGger:NEDGe:TIMe

- **Command Format**

  :TRIGger:NEDGe:TIMe <time>

  :TRIGger:NEDGe:TIMe?

- **Functional Description**

  Set the idle time for the Nth edge trigger.

- **Return Format**

  The query returns the current idle time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:NEDGe:TIMe 1 | Set the idle time for the Nth edge trigger to 1s. |
| :TRIGger:NEDGe:TIMe? | The query returns 1.000000e+00. |

### :TRIGger:NEDGe:EDGe

- **Command Format**

  :TRIGger:NEDGe:EDGe <value>

  :TRIGger:NEDGe:EDGe?

- **Functional Description**

  Set the N-edge count.

  <value>: Integer value, ranging from 1-65535.

- **Return Format**

  The query returns the current N-edge count.

- **For Example**

| | |
|---|---|
| :TRIGger:NEDGe:EDGe 100 | Set N-edge count to 100. |
| :TRIGger:NEDGe:EDGe? | The query returns 100. |

## Code Pattern Triggering

### :TRIGger:PATTern:LEVel

- **Command Format**

  :TRIGger:PATTern:LEVel<level>

  :TRIGger:PATTern:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:PATTern:LEVel 3 | Set the trigger level to 3 V. |
| :TRIGger:PATTern:LEVel? | The query returns 3.000000e+00. |

### :TRIGger:PATTern:PATTern

- **Command Format**

  :TRIGger:PATTern:PATTern <source>,<pch>

  :TRIGger:PATTern:PATTern? <source>

- **Functional Description**

  Set or query the trigger code pattern for the specified source. X represents the default value.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

  <pch>: {H|L|X|R|F}.

- **Return Format**

  The query returns the current trigger code pattern of the specified source.

- **For Example**

| | |
|---|---|
| :TRIGger:PATTern:PATTern CHANnel1,H | Set the code pattern of Channel 1 as H. |
| :TRIGger:PATTern:PATTern? CHANnel1 | The query returns H. |

## RS232 Triggering

### :TRIGger:RS232:SOURce

- **Command Format**

  :TRIGger:RS232:SOURce <source>

  :TRIGger:RS232:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:RS232:SOURce? | The query returns CHANnel1. |

### :TRIGger:RS232:LEVel

- **Command Format**

  :TRIGger:RS232:LEVel <level>

:TRIGger:RS232:LEVel?

- **Functional Description**

Set or query the trigger level value.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:LEVel 2 | Set the trigger level to 2 V. |
| :TRIGger:RS232:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:RS232:POLarity

- **Command Format**

:TRIGger:RS232:POLarity {POSitive|NEGative }

:TRIGger:RS232:POLarity?

- **Functional Description**

Set or query the trigger pulse polarity to POSitive (Positive) or NEGative (Negative).

- **Return Format**

The query returns the trigger pulse polarity { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:POLarity POS | Set the trigger pulse polarity to POSitive (Positive). |
| :TRIGger:RS232:POLarity? | The query returns POSitive. |

### :TRIGger:RS232:QUALifier

- **Command Format**

:TRIGger:RS232:QUALifier {STARt|ERRor|CERRor|DATA}

:TRIGger:RS232:QUALifier?

- **Functional Description**

Set or query the trigger condition.

STARt: Trigger at the start frame.

ERRor: Triggered when an error frame is detected.

CERRor: Triggered when a check error is detected.

DATA: Trigger at the last set data bit.

- **Return Format**

The query returns {STARt|ERRor|CERRor|DATA}.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:QUALifier STARt | Set the trigger condition to STARt. |
| :TRIGger:RS232:QUALifier? | The query returns STARt. |

### :TRIGger:RS232:ORDer

- **Command Format**

:TRIGger:RS232:ORDer {LSB|MSB}

:TRIGger:RS232:ORDer?

- **Functional Description**

Set or query the byte order for the RS232 bus trigger.

LSB: Least significant bit

MSB: Most significant bit

- **Return Format**

The query returns {LSB|MSB}.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:ORDer LSB | Set the byte order to LSB. |
| :TRIGger:RS232:ORDer? | The query returns LSB. |

### :TRIGger:RS232:BAUDrate

- **Command Format**

:TRIGger:RS232:BAUDrate <baud rate>

:TRIGger:RS232:BAUDrate?

- **Functional Description**

Set or query the baud rate for the RS232 trigger. The default unit is bps, and it is an integer.

- **Return Format**

The query returns the baud rate.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:BAUDrate 9600 | Set the baud rate for the RS232 trigger to 9600 bps. |
| :TRIGger:RS232:BAUDrate? | The query returns 9,600. |

### :TRIGger:RS232:WIDTh

- **Command Format**

:TRIGger:RS232:WIDTh {5|6|7|8}

:TRIGger:RS232:WIDTh?

- **Functional Description**

Set or query the data bit width for the RS232 trigger in the DATA trigger condition.

- **Return Format**

The query returns {5|6|7|8}.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:WIDTh 6 | Set the data bit width for RS232 trigger to 6. |
| :TRIGger:RS232:WIDTh? | The query returns 6. |

### :TRIGger:RS232:STOP

- **Command Format**

:TRIGger:RS232:STOP {1|2}

:TRIGger:RS232:STOP?

- **Functional Description**

Set or query the stop bit for the RS232 trigger.

- **Return Format**

The query returns {1|2}.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:STOP 1 | Set the stop bit for the RS232 trigger to 1. |
| :TRIGger:RS232:STOP? | The query returns 1. |

### :TRIGger:RS232:PARity

- **Command Format**

:TRIGger:RS232:PARity {EVEN | ODD | NONE}

:TRIGger:RS232:PARity?

- **Functional Description**

Set or query the parity check for the RS232 trigger.

- **Return Format**

The query returns {EVEN | ODD | NONE}.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:PARity ODD | Set the parity check for the RS232 trigger to ODD. |
| :TRIGger:RS232:PARity? | The query returns ODD. |

### :TRIGger:RS232:DATA

- **Command Format**

  :TRIGger:RS232:DATA <data>

  :TRIGger:RS232:DATA?

- **Functional Description**

  Set or query the data for the RS232 trigger in the trigger condition of DATA.

  <data>: The parameter is in binary format, where the values can be 0 or 1. The range of values is determined by the setting of the :TRIGger:RS232:WIDTh, ranging from 0 to 2^n-1, where n is the current data bit width.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:RS232:DATA "01111111" | Set the data value to 0x7F. |
| :TRIGger:RS232:DATA? | The query returns 01111111. |

## I²C Triggering

### :TRIGger:I2C:SDA

- **Command Format**

  :TRIGger:I2C:SDA <source>

  :TRIGger:I2C:SDA?

- **Functional Description**

  Set or query the data source for the I2C trigger.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:I2C:SDA CHANnel1 | Set the data source for the I2C trigger to Channel 1. |
| :TRIGger:I2C:SDA? | The query returns CHANnel1. |

### :TRIGger:I2C:SCL

- **Command Format**

  :TRIGger:I2C:SCL <source>

:TRIGger:I2C:SCL?

- **Functional Description**

Set or query the clock source for the I2C trigger.

<source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:I2C:SCL CHANnel1 | Set the clock source for the I2C trigger to Channel 1. |
| :TRIGger:I2C:SCL? | The query returns CHANnel1. |

### :TRIGger:I2C:DLEVel

- **Command Format**

:TRIGger:I2C:DLEVel <level>

:TRIGger:I2C:DLEVel?

- **Functional Description**

Set or query the trigger level of data line for the I2C trigger.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:I2C:DLEVel 2 | Set the trigger level of data line for the I2C trigger to 2 V. |
| :TRIGger:I2C:DLEVel? | The query returns 2.000000e+00. |

### :TRIGger:I2C:CLEVel

- **Command Format**

:TRIGger:I2C:CLEVel <level>

:TRIGger:I2C:CLEVel?

- **Functional Description**

Set or query the trigger level of clock line for the I2C trigger.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:I2C:CLEVel 2 | Set the trigger level of clock line for the I2C trigger to 2 V. |
| :TRIGger:I2C:CLEVel? | The query returns 2.000000e+00. |

### :TRIGger:I2C:DIRection

- **Command Format**

:TRIGger:I2C:DIRection { READ | WRITe }

:TRIGger:I2C:DIRection?

- **Functional Description**

Set and query the data direction for the I2C trigger in the trigger conditions of Address or Address and Data.

- **Return Format**

The query returns { READ | WRITe }.

- **For Example**

| | |
|---|---|
| :TRIGger:I2C:DIRection READ | Set the data direction to READ. |
| :TRIGger:I2C:DIRection? | The query returns READ. |

### :TRIGger:I2C:QUALifier

- **Command Format**

:TRIGger:I2C:QUALifier {STARt|RESTart|STOP|NACK|ADDRess|DATA|ADATA}

:TRIGger:I2C:QUALifier?

- **Functional Description**

Set and query the trigger condition for the I2C trigger.

STARt: Triggered when SCL is high and SDA transitions from high to low.

RESTart: Triggered when another start condition occurs before the stop condition.

STOP: Triggered when SCL is high and SDA transitions from low to high.

NACK: Triggered if SDA is high during the acknowledgment clock on SCL.

ADDRess: Triggered on read and write when the set address value is detected.

DATA: Triggered on the data line (SDA) when the set data value is found, at the clock edge of the last bit of data on the clock line (SCL).

ADATA: Triggered when both the set address value and data value are found, simultaneously satisfying the Address and Data conditions.

- **Return Format**

  The query returns {STARt|RESTart|STOP|NACK|ADDRess|DATA|ADATA}.

- **For Example**

  :TRIGger:I2C:QUALifier STOP Set the trigger condition for the I2C trigger to STOP.

  :TRIGger:I2C:QUALifier? The query returns STOP.

### :TRIGger:I2C:AWIDth

- **Command Format**

  :TRIGger:I2C:AWIDTh {7 | 10}

  :TRIGger:I2C:AWIDTh?

- **Functional Description**

  Set or query the address bit width for the I2C trigger in the trigger conditions of Address or Address and Data.

- **Return Format**

  The query returns {7 | 10}.

- **For Example**

  :TRIGger:I2C:AWIDTh 7 Set the address bit width to 7.

  :TRIGger:I2C:AWIDTh? The query returns 7.

### :TRIGger:I2C:ADDRess

- **Command Format**

  :TRIGger:I2C:ADDRess <address>

  :TRIGger:I2C:ADDRess?

- **Functional Description**

  Set or query the address value for the I2C trigger in the trigger conditions of Address or Address and Data.

  <address>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:I2C:AWIDth, ranging from 0-2^n - 1. Here, n equals the current address width.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

  :TRIGger:I2C:ADDRess "X0X00X1" Set the address value to X0X00X1.

| | |
|---|---|
| :TRIGger:I2C:ADDRess? | The query returns X0X00X1. |

### :TRIGger:I2C:DBYTes

- **Command Format**

  :TRIGger:I2C:DBYTes <len>

  :TRIGger:I2C:DBYTes?

- **Functional Description**

  Set or query the data length for the I2C trigger in the trigger conditions of Address or Address and Data.

  <len>: Data length, ranging from 1 to 5.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:I2C:DBYTes 2 | Set the data length to 2. |
| :TRIGger:I2C:DBYTes? | The query returns 2. |

### :TRIGger:I2C:DATA

- **Command Format**

  :TRIGger:I2C:DATA <data>

  :TRIGger:I2C:DATA?

- **Functional Description**

  Set or query the data value for the I2C trigger in the trigger conditions of Address or Address and Data.

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:I2C:DBYTes, ranging from 0-2^n - 1. Here, n equals the current data length multiplied by 8.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:I2C:DATA "X00X00X1" | Set the data value to X00X00X1. |
| :TRIGger:I2C:DATA? | The query returns X00X00X1. |

## SPI Triggering

### :TRIGger:SPI:SCL

- **Command Format**

  :TRIGger:SPI:SCL <source>

  :TRIGger:SPI:SCL?

- **Functional Description**

  Set or query the clock source for the SPI trigger.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:SCL CHANnel1 | Set the clock source for the SPI trigger to Channel 1. |
| :TRIGger:SPI:SCL? | The query returns CHANnel1. |

### :TRIGger:SPI:SMOSI

- **Command Format**

  :TRIGger:SPI:SMOSI <source>

  :TRIGger:SPI:SMOSI?

- **Functional Description**

  Set or query the MOSI source for the SPI trigger.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:SMOSI CHANnel1 | Set the MOSI source for the SPI trigger to Channel 1. |
| :TRIGger:SPI:SMOSI? | The query returns CHANnel1. |

### :TRIGger:SPI:SCS

- **Command Format**

  :TRIGger:SPI:SCS <source>

:TRIGger:SPI:SCS?

- **Functional Description**

Set or query the chip select source for the SPI trigger.

<source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:SCS CHANnel1 | Set the chip select source for the SPI trigger to Channel 1. |
| :TRIGger:SPI:SCS? | The query returns CHANnel1. |

### :TRIGger:SPI:CLOCk:LEVel

- **Command Format**

:TRIGger:SPI:CLOCk:LEVel <level>

:TRIGger:SPI:CLOCk:LEVel?

- **Functional Description**

Set or query the trigger level of the clock line for the SPI trigger.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:CLOCk:LEVel 2 | Set the trigger level of the clock line for the SPI trigger to 2 V. |
| :TRIGger:SPI:CLOCk:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:SPI:MOSI:LEVel

- **Command Format**

:TRIGger:SPI:MOSI:LEVel <level>

:TRIGger:SPI:MOSI:LEVel?

- **Functional Description**

Set or query the trigger level of MOSI data line for the SPI trigger.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

:TRIGger:SPI:MOSI:LEVel 2 Set the trigger level of MOSI data line for the SPI trigger to 2 V.

:TRIGger:SPI:MOSI:LEVel? The query returns 2.000000e+00.

### :TRIGger:SPI:CS:LEVel

- **Command Format**

:TRIGger:SPI:CS:LEVel <level>

:TRIGger:SPI:CS:LEVel?

- **Functional Description**

Set or query the trigger level of chip select line for the SPI trigger.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

:TRIGger:SPI:CS:LEVel 2 Set the trigger level of chip select line for the SPI trigger to 2 V.

:TRIGger:SPI:CS:LEVel? The query returns 2.000000e+00.

### :TRIGger:SPI:CLOCk:POLarity

- **Command Format**

:TRIGger:SPI:CLOCk:POLarity {POSitive|NEGative}

:TRIGger:SPI:CLOCk:POLarity?

- **Functional Description**

Set or query the pulse polarity of the clock line for the SPI trigger to POSitive (Positive) or NEGative (Negative).

- **Return Format**

The query returns the pulse polarity of the trigger { POSitive | NEGative}.

- **For Example**

:TRIGger:SPI:CLOCk:POLarity POS Set the pulse polarity of the clock line for the SPI trigger to POSitive (Positive).

:TRIGger:SPI:CLOCk:POLarity? The query returns POSitive.

### :TRIGger:SPI:MOSI:POLarity

- **Command Format**

  :TRIGger:SPI:MOSI:POLarity {POSitive|NEGative }

  :TRIGger:SPI:MOSI:POLarity?

- **Functional Description**

  Set or query the pulse polarity of the MOSI data line for the SPI trigger to POSitive (Positive) or NEGative (Negative).

- **Return Format**

  The query returns the pulse polarity of the trigger { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:MOSI:POLarity POS | Set the pulse polarity of the MOSI data line for the SPI trigger to POSitive (Positive). |
| :TRIGger:SPI:MOSI:POLarity? | The query returns POSitive. |

### :TRIGger:SPI:CS:POLarity

- **Command Format**

  :TRIGger:SPI:CS:POLarity {POSitive|NEGative }

  :TRIGger:SPI:CS:POLarity?

- **Functional Description**

  Set or query the pulse polarity of the chip select line for the SPI trigger to POSitive (Positive) or NEGative (Negative).

- **Return Format**

  The query returns the pulse polarity of the trigger { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:CS:POLarity POS | Set the pulse polarity of the chip select line for the SPI trigger to POSitive (Positive). |
| :TRIGger:SPI:CS:POLarity? | The query returns POSitive. |

### :TRIGger:SPI:QUALifier

- **Command Format**

  :TRIGger:SPI:QUALifier {STARt|DATA}

  :TRIGger:SPI:QUALifier?

- **Functional Description**

  Set and query the trigger condition for the SPI trigger.

STARt: Start

DATA: Data

- **Return Format**

The query returns {STARt|DATA}.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:QUALifier DATA | Set the trigger condition for the SPI trigger to DATA. |
| :TRIGger:SPI:QUALifier? | The query returns DATA. |

### :TRIGger:SPI:DWIDth

- **Command Format**

:TRIGger:SPI:DWIDth <width>

:TRIGger:SPI:DWIDth?

- **Functional Description**

Set or query the data bit width for the SPI trigger in the trigger conditions of Idle Data or CS Data.

<width>: Data bit width, ranging from 4 to 32.

- **Return Format**

The query returns the data bit width as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:DWIDth 4 | Set the data bit width to 4. |
| :TRIGger:SPI:DWIDth? | The query returns 4. |

### :TRIGger:SPI:DLENgth

- **Command Format**

:TRIGger:SPI:DLENgth <len>

:TRIGger:SPI:DLENgth?

- **Functional Description**

Set or query the data frame length for the SPI trigger in the trigger conditions of Idle Data or CS Data.

<len>: Data frame length, ranging from 1 to 32.

- **Return Format**

The query returns the data frame length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:DLENgth 4 | Set the data frame length to 4. |

:TRIGger:SPI:DLENgth? The query returns 4.

### :TRIGger:SPI:DATA

- **Command Format**

  :TRIGger:SPI:DATA <data>

  :TRIGger:SPI:DATA?

- **Functional Description**

  Set or query the data value for the SPI trigger in the trigger conditions of Idle Data or CS Data.

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the commands :TRIGger:SPI:DWIDth and :TRIGger:SPI:DLENgth, ranging from 0-2^n - 1. Here, n equals the current data length multiplied by data bit width.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

  :TRIGger:SPI:DATA "X00X00X1" Set the data value to X00X00X1.

  :TRIGger:SPI:DATA? The query returns X00X00X1.

### :TRIGger:SPI:MODe

- **Command Format**

  :TRIGger:SPI:MODe { CS | TIMeout }

  :TRIGger:SPI:MODe?

- **Functional Description**

  Set or query the bus mode for the SPI trigger.

- **Return Format**

  The query returns { CS | TIMeout }.

- **For Example**

  :TRIGger:SPI:MODe TIMeout Set the bus mode for the SPI trigger to TIMeout.

  :TRIGger:SPI:MODe? The query returns TIMeout.

### :TRIGger:SPI:TIMe

- **Command Format**

  :TRIGger:SPI:TIMe <time>

  :TRIGger:SPI:TIMe?

- **Functional Description**

  Set or query the idle time for the SPI trigger in timeout mode.

- **Return Format**

  The query returns the current idle time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :TRIGger:SPI:TIMe 1 | Set the idle time for the SPI trigger in timeout mode to 1s. |
| :TRIGger:SPI:TIMe? | The query returns 1.000000e+00. |

## CAN Triggering

### :TRIGger:CAN:SOURce

- **Command Format**

  :TRIGger:CAN:SOURce <source>

  :TRIGger:CAN:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:CAN:SOURce? | The query returns CHANnel1. |

### :TRIGger:CAN:LEVel

- **Command Format**

  :TRIGger:CAN:LEVel <level>

  :TRIGger:CAN:LEVel?

- **Functional Description**

  Set or query the trigger level value.

  <level>: Trigger level value

- **Return Format**

  The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:LEVel 2 | Set the trigger level to 2 V. |
| :TRIGger:CAN:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:CAN:STYPe

- **Command Format**

:TRIGger:CAN:STYPe { L | H }

:TRIGger:CAN:STYPe?

- **Functional Description**

Set or query the signal type for the CAN trigger.

H: Actual CAN_H bus signal

L: Actual CAN_L bus signal

- **Return Format**

The query returns { L | H }.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:STYPe H | Set the signal type for the CAN trigger to CAN_H. |
| :TRIGger:CAN:STYPe? | The query returns H. |

### :TRIGger:CAN:QUALifier

- **Command Format**

:TRIGger:CAN:QUALifier

{SOF|DFRame|REMote|ERRor|OVERload|ID|DATA|IDData|EOF|ACK|ERBit|CRCERRor|ALLERRor}

:TRIGger:CAN:QUALifier?

- **Functional Description**

Set or query the trigger condition for the CAN trigger.

SOF: Start of frame

DFRame: Data frame

REMote: Remote frame

ERRor: Error frame

OVERload: Overload frame

ID: Identifier

DATA: Data

IDData: ID and Data

EOF: End of frame

ACK: Loss of acknowledgement

ERBit: Bit stuffing error

CRCERRor: Cyclic redundancy check error

ALLERRor: All error

- **Return Format**

The query returns

{SOF|DFRame|REMote|ERRor|OVERload|ID|DATA|IDData|EOF|ACK|ERBit|CRCERRor|ALLERRor}.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:QUALifier SOF | Set the trigger condition for the CAN trigger to SOF. |
| :TRIGger:CAN:QUALifier? | The query returns SOF. |

### :TRIGger:CAN:BAUDrate

- **Command Format**

:TRIGger:CAN:BAUDrate <baud rate>

:TRIGger:CAN:BAUDrate?

- **Functional Description**

Set or query the signal baud rate for the CAN trigger.

<baud rate>: Baud rate, ranging from 10,000 to 1,000,000 bps.

- **Return Format**

The query returns the baud rate as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:BAUDrate 100000 | Set the signal baud rate for the CAN trigger to 100 kbps. |
| :TRIGger:CAN:BAUDrate? | The query returns 100,000. |

### :TRIGger:CAN:IDFormat

- **Command Format**

:TRIGger:CAN:IDFormat {STANdard | EXTended}

:TRIGger:CAN:IDFormat?

- **Functional Description**

Set or query the ID (Identifier) format for the CAN trigger.

- **Return Format**

The query returns {STANdard | EXTended}.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:IDFormat STANdard | Set the ID (Identifier) format for the CAN trigger to |

| | |
|---|---|
| | STANdard. |
| :TRIGger:CAN:IDFormat? | The query returns STANdard. |

### :TRIGger:CAN:IDDirection

- **Command Format**

  :TRIGger:CAN:IDDirection { READ | WRITE | ANY}

  :TRIGger:CAN:IDDirection?

- **Functional Description**

  Set or query the data direction for the CAN trigger in the trigger condition of ID (Identifier).

- **Return Format**

  The query returns { READ | WRITE | ANY}.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:IDDIRection READ | Set the data direction to READ. |
| :TRIGger:CAN:IDDIRection? | The query returns READ. |

### :TRIGger:CAN:ID

- **Command Format**

  :TRIGger:CAN:ID <data>

  :TRIGger:CAN:ID?

- **Functional Description**

  Set or query the data for the CAN trigger in the trigger condition of ID (Identifier).

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:CAN:IDFormat. The standard frame range is 0x0-07FF, data bit takes 11 bits; the extend frame range is 0x0-0x1FFFFFFF, data bit takes 29 bits, ranging from 0-2^n - 1. Here, n equals the current data length multiplied by data bit width.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:ID "000X00X00X1" | Set the data of ID (Identifier) to 000X00X00X1. |
| :TRIGger:CAN:ID? | The query returns 000X00X00X1. |

### :TRIGger:CAN:DLENgth

- **Command Format**

:TRIGger:CAN:DLENgth <len>

:TRIGger:CAN:DLENgth?

- **Functional Description**

Set or query the data length for the CAN trigger in the trigger conditions of data or ID and data.

<len>: Data length, ranging from 1 to 8.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:DLENgth 4 | Set the data length of the data bits to 4. |
| :TRIGger:CAN:DLENgth? | The query returns 4. |

### :TRIGger:CAN:DATA

- **Command Format**

:TRIGger:CAN:DATA <data>

:TRIGger:CAN:DATA?

- **Functional Description**

Set or query the data for the CAN trigger in the trigger conditions of data or ID and data.

<data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:CAN:DLENgth, ranging from 0-2^n-1. Here, n equals the current data length multiplied by 8.

- **Return Format**

The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:CAN:DATA "X00X00X1" | Set the data value to X00X00X1. |
| :TRIGger:CAN:DATA? | The query returns X00X00X1. |

## CAN-FD Triggering

### :TRIGger:CANFD:SOURce

- **Command Format**

:TRIGger:CANFD:SOURce <source>

:TRIGger:CANFD:SOURce?

- **Functional Description**

Set or query the trigger source.

<source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:CANFD:SOURce? | The query returns CHANnel1. |

### :TRIGger:CANFD:LEVel

- **Command Format**

:TRIGger:CANFD:LEVel <level>

:TRIGger:CANFD:LEVel?

- **Functional Description**

Set or query the trigger level value.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:LEVel 2 | Set the trigger level to 2 V. |
| :TRIGger:CANFD:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:CANFD:STYPe

- **Command Format**

:TRIGger:CANFD:STYPe { L | H }

:TRIGger:CANFD:STYPe?

- **Functional Description**

Set or query the signal type for the CAN-FD trigger.

H: Actual CAN_H bus signal

L: Actual CAN_L bus signal

- **Return Format**

The query returns { L | H }.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:STYPe H | Set the signal type for the CAN-FD trigger to CAN_H. |
| :TRIGger:CANFD:STYPe? | The query returns H. |

### :TRIGger:CANFD:QUALifier

- **Command Format**

:TRIGger:CANFD:QUALifier {SOF|DFRame|REMote|ERRor|OVERload|ID|DATA|IDData|EOF|ACK|ERBit|CRCERRor|ALLERRor}

:TRIGger:CANFD:QUALifier?

- **Functional Description**

Set or query the trigger condition for the CAN-FD trigger.

SOF: Start of frame

DFRame: Data frame

REMote: Remote frame

ERRor: Error frame

OVERload: Overload frame

ID: Identifier

DATA: Data

IDData: ID and Data

EOF: End of frame

ACK: Loss of acknowledgement

ERBit: Bit stuffing error

CRCERRor: Cyclic redundancy check error

ALLERRor: All error

- **Return Format**

The query returns {SOF|DFRame|REMote|ERRor|OVERload|ID|DATA|IDData|EOF|ACK|ERBit|CRCERRor|ALLERRor}.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:QUALifier SOF | Set the trigger condition for the CAN-FD trigger to SOF. |
| :TRIGger:CANFD:QUALifier? | The query returns SOF. |

### :TRIGger:CANFD:BAUDrate

- **Command Format**

:TRIGger:CANFD:BAUDrate <baud rate>

:TRIGger:CANFD:BAUDrate?

- **Functional Description**

  Set or query the signal baud rate for the CAN-FD trigger.

  <baud rate>: Baud rate, ranging from 10,000 to 1,000,000 bps.

- **Return Format**

  The query returns the baud rate as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:BAUDrate 100000 | Set the signal baud rate for the CAN-FD trigger to 100 kbps. |
| :TRIGger:CANFD:BAUDrate? | The query returns 100,000. |

### :TRIGger:CANFD:FD:BAUDrate

- **Command Format**

  :TRIGger:CANFD:FD:BAUDrate <baud rate>

  :TRIGger:CANFD:FD:BAUDrate?

- **Functional Description**

  Set or query the FD baud rate for the CAN-FD trigger.

  <baud rate>: Baud rate, ranging from 250,000 to 8,000,000 bps.

- **Return Format**

  The query returns the baud rate as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:FD:BAUDrate 500000 | Set the FD baud rate for the CAN-FD trigger to 500 kbps. |
| :TRIGger:CANFD:FD:BAUDrate? | The query returns 500,000. |

### :TRIGger:CANFD:SPOSition

- **Command Format**

  :TRIGger:CANFD:SPOSition <position>

  :TRIGger:CANFD:SPOSition?

- **Functional Description**

  Set or query the signal sampling position for the CAN-FD trigger.

  <position>: Sampling position, ranging from 30 to 90, with the unit in %.

- **Return Format**

  The query returns the signal sampling position in scientific notation.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:SPOSition 40 | Set the signal sampling position for the CAN-FD trigger to 40%. |
| :TRIGger:CANFD:SPOSition? | The query returns 4000000e+01. |

### :TRIGger:CANFD:IDFormat

- **Command Format**

  :TRIGger:CANFD:IDFormat {STANdard | EXTended | FDSTandard | FDEXtended}

  :TRIGger:CANFD:IDFormat?

- **Functional Description**

  Set or query the ID (Identifier) format for the CAN-FD trigger.

- **Return Format**

  The query returns {STANdard | EXTended | FDSTandard | FDEXtended}.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:IDFormat STANdard | Set the ID (Identifier) format to STANdard. |
| :TRIGger:CANFD:IDFormat? | The query returns STANdard. |

### :TRIGger:CANFD:ID

- **Command Format**

  :TRIGger:CANFD:ID <data>

  :TRIGger:CANFD:ID?

- **Functional Description**

  Set or query the data for the CAN-FD trigger in the trigger conditions of ID (Identifier).

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:CANFD:IDFormat. The standard frame range is 0x0-07FF, data bit takes 11 bits; the extend frame range is 0x0-0x1FFFFFFF, data bit takes 29 bits; FD standard frame range is 0x0-0x7FF, data bit takes 11 bits; FD extend frame range is 0x0-0x1FFFFFFF, data bit takes 29 bits.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:ID "000X00X00X1" | Set the data for ID (Identifier) to 000X00X00X1. |
| :TRIGger:CANFD:ID? | The query returns 000X00X00X1. |

### :TRIGger:CANFD:DLENgth

- **Command Format**

  :TRIGger:CANFD:DLENgth <len>

  :TRIGger:CANFD:DLENgth?

- **Functional Description**

  Set or query the data length for the CAN-FD trigger in the trigger conditions of data or ID and data.

  <len>: Data length, ranging from 1 to 16.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:DLENgth 4 | Set the data length of the data bits to 4. |
| :TRIGger:CANFD:DLENgth? | The query returns 4. |

### :TRIGger:CANFD:DATA

- **Command Format**

  :TRIGger:CANFD:DATA <data>

  :TRIGger:CANFD:DATA?

- **Functional Description**

  Set or query the data for the CAN-FD trigger in the trigger conditions of data or ID and data.

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:CANFD:DLENgth, ranging from 0-2^n-1. Here, n equals the current data length multiplied by 8.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:CANFD:DATA "X00X00X1" | Set the data value to X00X00X1. |
| :TRIGger:CANFD:DATA? | The query returns X00X00X1. |

### :TRIGger:CANFD:DATA:OFFSet:CTL

- **Command Format**

  :TRIGger:CANFD:DATA:OFFSet:CTL { {1|ON} | {0|OFF} }

  :TRIGger:CANFD:DATA:OFFSet:CTL?

- **Functional Description**

  Set or query the switch state of byte bias for the CAN-FD trigger in the trigger conditions of data or ID and data.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

  :TRIGger:CANFD:DATA:OFFSet:CTL ON Enable data byte offset.

  :TRIGger:CANFD:DATA:OFFSet:CTL? The query returns 1.

### :TRIGger:CANFD:DATA:OFFSet

- **Command Format**

  :TRIGger:CANFD:DATA:OFFSet <offset>

  :TRIGger:CANFD:DATA:OFFSet?

- **Functional Description**

  Set or query the data byte offset for the CAN-FD trigger in the trigger conditions of data or ID and data. When using this command, it is enabled by default.

  <offset>: Byte bias, ranging from 0 to 63.

- **Return Format**

  The query returns the data byte offset as an integer.

- **For Example**

  :TRIGger:CANFD:DATA:OFFSet 8 Set the data byte offset for the CAN-FD trigger to 8.

  :TRIGger:CANFD:DATA:OFFSet? The query returns 8.

## LIN Triggering

### :TRIGger:LIN:SOURce

- **Command Format**

  :TRIGger:LIN:SOURce <source>

  :TRIGger:LIN:SOURce?

- **Functional Description**

  Set or query the trigger source.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:LIN:SOURce? | The query returns CHANnel1. |

### :TRIGger:LIN:LEVel

- **Command Format**

:TRIGger:LIN:LEVel <level>

:TRIGger:LIN:LEVel?

- **Functional Description**

Set or query the trigger level value.

<level>: Trigger level value

- **Return Format**

The query returns the trigger level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:LEVel 2 | Set the trigger level to 2 V. |
| :TRIGger:LIN:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:LIN:POLarity

- **Command Format**

:TRIGger:LIN:POLarity {NORMal | INVert}

:TRIGger:LIN:POLarity?

- **Functional Description**

Set the polarity for the LIN bus trigger.

NORMal: Normal, high=1

INVert: Invert, high=0

- **Return Format**

The query returns {NORMal | INVert}.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:POLarity NORMal | Set the polarity to NORMal. |
| :TRIGger:LIN:POLarity? | The query returns NORMal. |

### :TRIGger:LIN:QUALifier

- **Command Format**

  :TRIGger:LIN:QUALifier {SYNC|ID|DATA|IDData|WAKe|SLEep|ERRor}

  :TRIGger:LIN:QUALifier?

- **Functional Description**

  Set or query the trigger condition for the LIN trigger.

  SYNC: Synchronization

  ID: Identifier

  DATA: Data

  IDData: ID and Data

  WAKe: Wake-up frame

  SLEep: Sleep frame

  ERRor: Error frame

- **Return Format**

  The query returns {SYNC|ID|DATA|IDData|WAKe|SLEep|ERRor}.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:QUALifier SYNC | Set the trigger condition to SYNC. |
| :TRIGger:LIN:QUALifier? | The query returns SYNC. |

### :TRIGger:LIN:VERSion

- **Command Format**

  :TRIGger:LIN:VERSion {VER1|VER2|ANY}

  :TRIGger:LIN:VERSion?

- **Functional Description**

  Set or query the version of the LIN bus trigger.

  {VER1| VER2|ANY} represents V1.x, V2.x, and any arbitrary version, respectively.

- **Return Format**

  The query returns {VER1|VER2|ANY}.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:VERSion VER1 | Set the version to VER1. |
| :TRIGger:LIN:VERSion? | The query returns VER1. |

### :TRIGger:LIN:BAUDrate

- **Command Format**

:TRIGger:LIN:BAUDrate <baud rate>

:TRIGger:LIN:BAUDrate?

- **Functional Description**

Set or query the signal baud rate for the LIN trigger.

<baud rate>: Baud rate, ranging from 1,200 to 1,000,000 bps.

- **Return Format**

The query returns the baud rate as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:BAUDrate 2400 | Set the signal baud rate for the LIN trigger to 2.4 kbps. |
| :TRIGger:LIN:BAUDrate? | The query returns 2, 400. |

### :TRIGger:LIN:ID:PARity

- **Command Format**

:TRIGger:LIN:ID:PARity { {1|ON} | {0|OFF} }

:TRIGger:LIN:ID:PARity?

- **Functional Description**

Set or query whether the ID of the LIN trigger includes the parity bit. ON indicates Yes, while OFF indicates No.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:ID:PARity ON | Set the ID include the parity bit. |
| :TRIGger:LIN:ID:PARity? | The query returns 1. |

### :TRIGger:LIN:LENGth:CTL

- **Command Format**

:TRIGger:LIN:LENGth:CTL { {1|ON} | {0|OFF} }

:TRIGger:LIN:LENGth:CTL?

- **Functional Description**

Set or query whether the data length of the LIN trigger is to be set. ON indicates Yes, while OFF indicates No.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:LENGth:CTL ON | Enable the data length. |
| :TRIGger:LIN:LENGth:CTL? | The query returns 1. |

### :TRIGger:LIN:LENGth

- **Command Format**

  :TRIGger:LIN:LENGth <length>

  :TRIGger:LIN:LENGth?

- **Functional Description**

  Set or query the data length for the LIN trigger. When using this command, it is enabled by default.

  <length>: Data length, ranging from 1 to 8.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:LENGth 6 | Set the data length to 6. |
| :TRIGger:LIN:LENGth? | The query returns 6. |

### :TRIGger:LIN:ERRor

- **Command Format**

  :TRIGger:LIN:ERRor {SYNC|ID|CHECk}

  :TRIGger:LIN:ERRor?

- **Functional Description**

  Set or query the error type for the LIN trigger in the trigger condition of Error.

  { SYNC | ID | CHECk} represents Sync, ID parity check, and checksum, respectively.

- **Return Format**

  The query returns {SYNC|ID|CHECk}.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:ERRor SYNC | Set the error type to SYNC. |
| :TRIGger:LIN:ERRor? | The query returns SYNC. |

### :TRIGger:LIN:ID

- **Command Format**

  :TRIGger:LIN:ID <data>

  :TRIGger:LIN:ID?

- **Functional Description**

  Set or query the data for the LIN trigger in the trigger conditions of ID or ID and Data.

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X indicates uncertainty, ranging from 0-2^8-1.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:ID "X00X00X1" | Set the data to X00X00X1. |
| :TRIGger:LIN:ID? | The query returns X00X00X1. |

### :TRIGger:LIN:DLENgth

- **Command Format**

  :TRIGger:LIN:DLENgth <len>

  :TRIGger:LIN:DLENgth?

- **Functional Description**

  Set or query the data length for the LIN trigger in the trigger conditions of ID or ID and Data.

  <len>: Data length, ranging from 1 to 8.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:DLENgth 4 | Set the data length of the data bits to 4. |
| :TRIGger:LIN:DLENgth? | The query returns 4. |

### :TRIGger:LIN:DATA

- **Command Format**

  :TRIGger:LIN:DATA <data>

  :TRIGger:LIN:DATA?

- **Functional Description**

  Set or query the data for the LIN trigger in the trigger conditions of ID or ID and Data.

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:LIN:DLENgth, ranging from 0-2^n - 1. Here, n equals the current data length multiplied by 8.

- **Return Format**

The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:LIN:DATA "X00X00X1" | Set the data value to X00X00X1. |
| :TRIGger:LIN:DATA? | The query returns X00X00X1. |

## AUDIO Triggering

### :TRIGger:AUDio:FORMat

- **Command Format**

:TRIGger:AUDio:FORMat {STANdard|MSB|LSB|TDM}

:TRIGger:AUDio:FORMat?

- **Functional Description**

Set or query the data format for the AUDIO trigger.

{STANdard|MSB|LSB|TDM} represents Standard, Left-justified, Right-justified, and Time division multiplexing, respectively.

- **Return Format**

The query returns {STANdard|MSB|LSB|TDM}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:FORMat STANdard | Set the data format to STANdard. |
| :TRIGger:AUDio:FORMat? | The query returns STANdard. |

### :TRIGger:AUDio:ORDer

- **Command Format**

:TRIGger:AUDio:ORDer {LSB|MSB}

:TRIGger:AUDio:ORDer?

- **Functional Description**

Set or query the byte order for the AUDIO trigger.

LSB: Least significant bit

MSB: Most significant bit

- **Return Format**

The query returns {LSB|MSB}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:ORDer LSB | Set the byte order to LSB. |
| :TRIGger:AUDio:ORDer? | The query returns LSB. |

### :TRIGger:AUDio:BCLock:SOURce

- **Command Format**

  :TRIGger:AUDio:BCLock:SOURce <source>

  :TRIGger:AUDio:BCLock:SOURce?

- **Functional Description**

  Set or query the trigger source for the bit clock.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:BCLock:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:AUDio:BCLock:SOURce? | The query returns CHANnel1. |

### :TRIGger:AUDio:WSELect:SOURce

- **Command Format**

  :TRIGger:AUDio:WSELect:SOURce <source>

  :TRIGger:AUDio:WSELect:SOURce?

- **Functional Description**

  Set or query the trigger source for word selection.

  <source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

  Explanation: CHANnel<n> indicates the physical channel, where n can take a value from 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:WSELect:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:AUDio:WSELect:SOURce? | The query returns CHANnel1. |

### :TRIGger:AUDio:DATA:SOURce

- **Command Format**

  :TRIGger:AUDio:DATA:SOURce <source>

:TRIGger:AUDio:DATA:SOURce?

- **Functional Description**

Set or query the trigger source of the data.

<source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:DATA:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:AUDio:DATA:SOURce? | The query returns CHANnel1. |

### :TRIGger:AUDio:FSYNc:SOURce

- **Command Format**

:TRIGger:AUDio:FSYNc:SOURce <source>

:TRIGger:AUDio:FSYNc:SOURce?

- **Functional Description**

Set or query the trigger source of frame sync in TDM format.

<source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the trigger source {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:FSYNc:SOURce CHANnel1 | Set the trigger source as Channel 1. |
| :TRIGger:AUDio:FSYNc:SOURce? | The query returns CHANnel1. |

### :TRIGger:AUDio:BCLock:LEVel

- **Command Format**

:TRIGger:AUDio:BCLock:LEVel <level>

:TRIGger:AUDio:BCLock:LEVel?

- **Functional Description**

Set or query the trigger threshold for the bit clock.

<level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:BCLock:LEVel 2 | Set the threshold to 2 V. |
| :TRIGger:AUDio:BCLock:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:AUDio:WSELect:LEVel

- **Command Format**

  :TRIGger:AUDio:WSELect:LEVel <level>

  :TRIGger:AUDio:WSELect:LEVel?

- **Functional Description**

  Set or query the word selection for the bit clock.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:WSELect:LEVel 2 | Set the threshold to 2 V. |
| :TRIGger:AUDio:WSELect:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:AUDio:DATA:LEVel

- **Command Format**

  :TRIGger:AUDio:DATA:LEVel <level>

  :TRIGger:AUDio:DATA:LEVel?

- **Functional Description**

  Set or query the trigger threshold for the bit data.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:DATA:LEVel 2 | Set the threshold to 2 V. |
| :TRIGger:AUDio:DATA:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:AUDio:FSYNc:LEVel

- **Command Format**

  :TRIGger:AUDio:FSYNc:LEVel <level>

  :TRIGger:AUDio:FSYNc:LEVel?

- **Functional Description**

  Set or query the trigger threshold of frame sync in TDM format.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:FSYNc:LEVel 2 | Set the threshold to 2 V. |
| :TRIGger:AUDio:FSYNc:LEVel? | The query returns 2.000000e+00. |

### :TRIGger:AUDio:BCLock:POLarity

- **Command Format**

  :TRIGger:AUDio:BCLock:POLarity {POSitive|NEGative}

  :TRIGger:AUDio:BCLock:POLarity?

- **Functional Description**

  Set or query the edge type of bit clock for the AUDIO trigger to POSitive (Rising) or NEGative (Falling).

- **Return Format**

  The query returns the trigger edge type { POSitive | NEGative }.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:BCLock:POLarity POS | Set the edge type of bit clock to POSitive (Rising edge). |
| :TRIGger:AUDio:BCLock:POLarity? | The query returns POSitive. |

### :TRIGger:AUDio:WSELect:POLarity

- **Command Format**

  :TRIGger:AUDio:WSELect:POLarity {NORMal|INVert}

  :TRIGger:AUDio:WSELect:POLarity?

- **Functional Description**

  Set or query the polarity of word selection for the AUDIO trigger to NORMal (Normal) or INVert

(Invert).

- **Return Format**

The query returns the trigger edge type {NORMal|INVert}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:WSELect:POLarity NORMal | Set the polarity of word selection to NORMal. |
| :TRIGger:AUDio:WSELect:POLarity? | The query returns NORMal. |

### :TRIGger:AUDio:DATA:POLarity

- **Command Format**

:TRIGger:AUDio:DATA:POLarity {H1|H0}

:TRIGger:AUDio:DATA:POLarity?

- **Functional Description**

Set or query the data polarity for the AUDIO trigger to H1 (H=1) or H0 (H=0).

- **Return Format**

The query returns the trigger edge type {H1|H0}.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:DATA:POLarity H1 | Set the data polarity to H=1. |
| :TRIGger:AUDio:DATA:POLarity? | The query returns H1. |

### :TRIGger:AUDio:FSYNc:POLarity

- **Command Format**

:TRIGger:AUDio:FSYNc:POLarity {POSitive|NEGative}

:TRIGger:AUDio:FSYNc:POLarity?

- **Functional Description**

Set or query the polarity of frame sync in TDM format to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns the trigger edge type { POSitive | NEGative }.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:FSYNc:POLarity POS | Set the polarity of frame sync to POSitive (Rising edge). |
| :TRIGger:AUDio:FSYNc:POLarity? | The query returns POSitive. |

### :TRIGger:AUDio:TYPe

- **Command Format**

  :TRIGger:AUDio:TYPe { WSELect | DATA }

  :TRIGger:AUDio:TYPe?

- **Functional Description**

  Set or query the trigger type for the AUDIO trigger in standard, left-justified, and right-justified formats.

  { WSELect | DATA } represents word selection and data, respectively.

- **Return Format**

  The query returns { WSELect | DATA }.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TYPe WSELect | Set the trigger type to WSELect. |
| :TRIGger:AUDio:TYPe? | The query returns WSELect. |

### :TRIGger:AUDio:VCHannel

- **Command Format**

  :TRIGger:AUDio:VCHannel { LEFT | RIGHt | ANY }

  :TRIGger:AUDio:VCHannel?

- **Functional Description**

  Set or query the sound channel for the AUDIO trigger in the standard, left-justified, and right-justified formats, and in data trigger mode.

  { LEFT | RIGHt | ANY } represents left channel, right channel, and either channel, respectively.

- **Return Format**

  The query returns { LEFT | RIGHt | ANY }.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:VCHannel LEFT | Set the sound channel to LEFT. |
| :TRIGger:AUDio:VCHannel? | The query returns LEFT. |

### :TRIGger:AUDio:DLENgth

- **Command Format**

  :TRIGger:AUDio:DLENgth <len>

  :TRIGger:AUDio:DLENgth?

- **Functional Description**

  Set or query the data length for the AUDIO trigger in standard, left-justified, and right-justified

formats.

<len>: Data length, ranging from 4 to 32.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:DLENgth 4 | Set the data length of the data bits to 4. |
| :TRIGger:AUDio:DLENgth? | The query returns 4. |

### :TRIGger:AUDio:DATA

- **Command Format**

:TRIGger:AUDio:DATA <data>

:TRIGger:AUDio:DATA?

- **Functional Description**

Set or query the data for the AUDIO trigger in standard, left-justified, and right-justified formats.

<data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:AUDio:DLENgth, ranging from 0-2^n-1. Here, n equals the length of the current data bit.

- **Return Format**

The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:DATA "X00X00X1" | Set the data value to X00X00X1. |
| :TRIGger:AUDio:DATA? | The query returns X00X00X1. |

### :TRIGger:AUDio:TDM:TYPe

- **Command Format**

:TRIGger:AUDio:TDM:TYPe { FSYNc | DATA | CDATA }

:TRIGger:AUDio:TDM:TYPe?

- **Functional Description**

Set or query the trigger type for the AUDIO trigger in TDM format.

{ FSYNc | DATA | CDATA } represents frame sync, data, and channel and data, respectively.

- **Return Format**

The query returns { FSYNc | DATA | CDATA }.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TDM:TYPe FSYNc | Set the trigger type to FSYNc. |
| :TRIGger:AUDio:TDM:TYPe? | The query returns FSYNc. |

### :TRIGger:AUDio:TDM:CLOCk

- **Command Format**

:TRIGger:AUDio:TDM:CLOCk <val>

:TRIGger:AUDio:TDM:CLOCk?

- **Functional Description**

Set or query the bit clock of each channel for the AUDIO trigger in TDM format.

<val>: Clock bit, ranging from 4 to 32.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TDM:CLOCk 4 | Set the clock bit to 4. |
| :TRIGger:AUDio:TDM:CLOCk? | The query returns 4. |

### :TRIGger:AUDio:TDM:NCHannel

- **Command Format**

:TRIGger:AUDio:TDM:NCHannel <val>

:TRIGger:AUDio:TDM:NCHannel?

- **Functional Description**

Set or query the channel number of each frame for the AUDIO trigger in TDM format.

<val>: Channel number, ranging from 2 to 64.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TDM:NCHannel 4 | Set the channel number to 4. |
| :TRIGger:AUDio:TDM:NCHannel? | The query returns 4. |

### :TRIGger:AUDio:TDM:DELay

- **Command Format**

:TRIGger:AUDio:TDM:DELay <val>

:TRIGger:AUDio:TDM:DELay?

- **Functional Description**

  Set or query the bit delay for the AUDIO trigger in TDM format.

  <val>: Bit delay, ranging from 0 to 31.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TDM:DELay 4 | Set the bit delay to 4. |
| :TRIGger:AUDio:TDM:DELay? | The query returns 4. |

### :TRIGger:AUDio:TDM:CHANnel

- **Command Format**

  :TRIGger:AUDio:TDM:CHANnel <val>

  :TRIGger:AUDio:TDM:CHANnel?

- **Functional Description**

  Set or query the channel for the AUDIO trigger in TDM format, and in channel and data trigger mode.

  <val>: Channel number, ranging from 0 to 63.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TDM:CHANnel 2 | Set the channel to 2. |
| :TRIGger:AUDio:TDM:CHANnel? | The query returns 2. |

### :TRIGger:AUDio:TDM:DLENgth

- **Command Format**

  :TRIGger:AUDio:TDM:DLENgth <len>

  :TRIGger:AUDio:TDM:DLENgth?

- **Functional Description**

  Set or query the data bit length of each channel for the AUDIO trigger in TDM format, and in channel and data trigger mode.

  <len>: Data length, ranging from 4 to 32.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TDM:DLENgth 4 | Set the data length of the data bits to 4. |
| :TRIGger:AUDio:TDM:DLENgth? | The query returns 4. |

### :TRIGger:AUDio:TDM:DATA

- **Command Format**

  :TRIGger:AUDio:TDM:DATA <data>

  :TRIGger:AUDio:TDM:DATA?

- **Functional Description**

  Set or query the data for the AUDIO trigger in TDM format, and in channel and data trigger mode.

  <data>: The parameter is in binary format, where the values can be 0, 1, or X. Here, X represents uncertainty. The range of values is determined by the setting of the command :TRIGger:AUDio:TDM:DLENgth, ranging from 0-2^n - 1. Here, n equals the length of the current data bit.

- **Return Format**

  The query returns the data string in binary format.

- **For Example**

| | |
|---|---|
| :TRIGger:AUDio:TDM:DATA "X00X00X1" | Set the data value to X00X00X1. |
| :TRIGger:AUDio:TDM:DATA? | The query returns X00X00X1. |

## CURSor Command

This command is used Set the cursor parameters for measuring the waveform data on the screen.

### :CURSor:MEASure

- **Command Format**

  :CURSor:MEASure {{1 | ON} | {0 | OFF}}

  :CURSor:MEASure?

- **Functional Description**

  Switch the cursor measurement to ON or OFF.

- **Return Format**

  The query returns the switch state of the cursor measurement: 1 indicates that the cursor measurement is enabled, while 0 indicates that the cursor measurement is disabled.

- **For Example**

| | |
|---|---|
| :CURSor:MEASure ON | Enable the cursor measurement. |

:CURSor:MEASure? The query returns 1.

### :CURSor:TYPe

- **Command Format**

:CURSor:TYPe {AMPlitude | TIMe | SCReen}

:CURSor:TYPe?

- **Functional Description**

Set the cursor type for the cursor measurement.

{AMPlitude | TIMe | SCReen} represents AMPlitude (Amplitude), TIMe (Time), and SCReen (Screen), respectively.

Explanation: In XY time base, only the amplitude and time cursors are available.

- **Return Format**

The query returns {AMPlitude | TIMe | SCReen }.

- **For Example**

:CURSor:TYPe AMP Set the cursor type to AMPlitude.

:CURSor:TYPe? The query returns AMPlitude.

### :CURSor:MODe

- **Command Format**

:CURSor:MODe{ TRACk | INDependent }

:CURSor:MODe?

- **Functional Description**

Set the cursor mode for the cursor measurement to TRACk (Track) or INDependent (Independent).

- **Return Format**

The query returns { TRACk | INDependent }.

- **For Example**

:CURSor:MODe TRACk Set the cursor mode to TRACk.

:CURSor:MODe? The query returns TRACk.

### :CURSor:X:MODe

- **Command Format**

:CURSor:X:MODe { POSition | DELay }

:CURSor:X:MODe?

- **Functional Description**

  Set or query the horizontal cursor mode.

  POSition represents the fixed position; the time difference of horizontal cursor line will change with the timebase.

  DELay represents the fixed delay; the time difference of horizontal cursor line will not change with the timebase.

- **Return Format**

  The query returns { POSition | DELay }.

- **For Example**

| | |
|---|---|
| :CURSor:X:MODe DELay | Set the horizontal cursor mode to DELay. |
| :CURSor:X:MODe? | The query returns DELay. |

## :CURSor

In YT mode, cursor measurements are based on a coordinate system with time as the X-axis and amplitude as the Y-axis.

### :CURSor:DISPlay

- **Command Format**

  :CURSor:DISPlay <sw>,<source>

  :CURSor:DISPlay? <source>

- **Functional Description**

  Set or query whether the source of cursor measurement is enabled.

  <sw> represents whether the source of cursor measurement is enabled: { {1|ON} | {0|OFF} }.

  <source> represents the source of cursor measurement: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns whether the source of cursor measurement is enabled: 0 indicates that the cursor measurement is disabled, while 1 indicates that the cursor measurement is enabled.

- **For Example**

| | |
|---|---|
| :CURSor:DISPlay ON,CHANnel1 | Enable the cursor measurement for Channel 1. |
| :CURSor:DISPlay OFF,CHANnel1 | Disable the cursor measurement for Channel 1. |
| :CURSor:DISPlay 1,CHANnel1 | Enable the cursor measurement for Channel 1. |
| :CURSor:DISPlay 0,CHANnel1 | Disable the cursor measurement for Channel 1. |
| :CURSor:DISPlay? CHANnel1 | The query returns 1, indicating that the cursor |

measurement of Channel 1 is enabled.

### :CURSor:CAX

- **Command Format**

  :CURSor:CAX <value>,<source>

  :CURSor:CAX? <source>

- **Functional Description**

  Set or query the horizontal position of cursor line A.

  <value>: Horizontal position, where the range of horizontal position is determined by the current horizontal time base and horizontal offset.

  <source> represents the source of cursor measurement: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the horizontal position of cursor line A in scientific notation, with the unit in seconds (s).

- **For Example**

  :CURSor:CAX 0.01,CHANnel1 Set the horizontal position for Channel 1's cursor A to 10 ms.

  :CURSor:CAX? CHANnel1 The query returns 1.000000e-02.

### :CURSor:CBX

- **Command Format**

  :CURSor:CBX <value>,<source>

  :CURSor:CBX? <source>

- **Functional Description**

  Set or query the horizontal position of cursor line B.

  <value>: Horizontal position, where the range of horizontal position is determined by the current horizontal time base and horizontal offset.

  <source> represents the source of cursor measurement: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the horizontal position of cursor line B in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :CURSor:CBX 0.01, CHANnel1 | Set the horizontal position for Channel 1's cursor B to 10 ms. |
| :CURSor:CBX? CHANnel1 | The query returns 1.000000e-02. |

### :CURSor:CAY

- **Command Format**

  :CURSor:CAY <value>,<source>

  :CURSor:CAY? <source>

- **Functional Description**

  Set or query the vertical position of cursor line A.

  <value>: Vertical position, where the range of vertical position is determined by the current vertical time base and vertical offset.

  <source> represents the source of cursor measurement: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the vertical position of cursor line A in scientific notation. The vertical unit is determined by the currently selected channel.

- **For Example**

| | |
|---|---|
| :CURSor:CAY 0.3, CHANnel1 | Set the vertical position for Channel 1's cursor A to 300 mV. |
| :CURSor:CAY? CHANnel1 | The query returns 3.000000e-01. |

### :CURSor:CBY

- **Command Format**

  :CURSor:CBY <value>,<source>

  :CURSor:CBY? <source>

- **Functional Description**

  Set or query the vertical position of cursor line B.

  <value>: Vertical position, where the range of vertical position is determined by the current vertical time base and vertical offset.

  <source> represents the source of cursor measurement: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the vertical position of cursor line B in scientific notation. The vertical unit is determined by the currently selected channel.

- **For Example**

:CURSor:CBY 0.3,CHANnel1 Set the vertical position for Channel 1's cursor B to 300 mV.

:CURSor:CBY? CHANnel1 he query returns 3.000000e-01.

### :CURSor:AXValue?

- **Command Format**

  :CURSor:AXValue? <source>

- **Functional Description**

  Query the X value at cursor A. The unit is determined by the horizontal unit of the currently selected channel.

  <source> represents the source of cursor measurement:

  {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the X value at cursor A in scientific notation.

- **For Example**

  :CURSor:AXV? CHANnel1 The query returns the X value at cursor A. for Channel 1 as 2.000000e-02.

### :CURSor:BXValue?

- **Command Format**

  :CURSor:BXValue? <source>

- **Functional Description**

  Query the X value at cursor B. The unit is determined by the horizontal unit of the currently selected channel.

  <source> represents the source of cursor measurement:

  {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the X value at Cursor B in scientific notation.

- **For Example**

  :CURSor:BXV? CHANnel1 The query returns the X value at cursor B. for Channel 1 as 2.000000e-02.

### :CURSor:AYValue?

- **Command Format**

  :CURSor:AYValue? <source>

- **Functional Description**

  Query the Y value at cursor A. The unit is determined by the vertical unit of the currently selected channel.

  <source> represents the source of cursor measurement: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

  In time measurement mode, the voltage at the intersection of cursor line A and the waveform of the corresponding channel in the YT coordinate system.

  In voltage or screen measurement mode, the Y value at cursor line A in the YT coordinate system.

- **Return Format**

  The query returns the Y value at cursor A in scientific notation.

- **For Example**

  :CURSor:AYV? CHANnel1 The query returns the Y value at cursor A for Channel 1 as 2.000000e-02.

**:CURSor:BYValue?**

- **Command Format**

  :CURSor:BYValue? <source>

- **Functional Description**

  Query the Y value at cursor B. The unit is determined by the vertical unit of the currently selected channel.

  <source> represents the source of cursor measurement: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

  In time measurement mode, the voltage at the intersection of cursor line B and the waveform of the corresponding channel in the YT coordinate system.

  In voltage or screen measurement mode, the Y value at cursor line B in the YT coordinate system.

- **Return Format**

  The query returns the Y value at Cursor B in scientific notation.

- **For Example**

  :CURSor:BYV? CHANnel1 The query returns the Y value at cursor B for Channel 1 as 2.000000e-02.

### :CURSor:XDELta?

- **Command Format**

  :CURSor:XDELta? <source>

- **Functional Description**

  Query the X difference "ΔX" between cursor A and cursor B. The unit is determined by the horizontal unit of the currently selected channel.

  <source> represents the source of cursor measurement:

  {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the X difference between cursor A and cursor B in scientific notation.

- **For Example**

  :CURSor:XDEL? CHANnel1 The query returns the X difference "ΔX" of Channel 1 as 2.000000e-02.

### :CURSor:YDELta?

- **Command Format**

  :CURSor:YDELta? <source>

- **Functional Description**

  Query the X difference "ΔY" between cursor A and cursor B. The unit is determined by the horizontal unit of the currently selected channel.

  <source> represents the source of cursor measurement:

  {CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4}.

- **Return Format**

  The query returns the Y difference between cursor A and cursor B in scientific notation.

- **For Example**

  :CURSor:YDEL? CHANnel1

  The query returns the X difference "ΔY" of Channel 1 as 2.000000e-01.

### :CURSor?

- **Command Format**

  :CURSor?

- **Functional Description**

  The query returns all measurement parameters at once in CSV format. The format conforms the Data Block Format.

- **Return Format**

  The query returns all measurement parameters in scientific notation, using standard units.

- **For Example**

  :CURSor? The query returns #9000000100

  NAME,A,B,DELTA,1/DELTA

  X,2.000000e-02,2.000000e-02,2.000000e-02,5.000000e+01

  Y C1,1.000000e-02,1.000000e-02,1.000000e-02,1.000000e+02

  X M1,1.000000e-02,1.000000e-02,1.000000e-02,1.000000e+02

  Y M1,1.000000e-02,1.000000e-02,1.000000e-02,1.000000e+02

  X M2,1.000000e-02,1.000000e-02,1.000000e-02,1.000000e+02

  …….....

## :CURSor:XY

In XY mode, cursor measurements are based on a coordinate system where the amplitudes of the two selected channels form the X and Y axes.

### :CURSor:XY:CAX

- **Command Format**

  :CURSor:XY:CAX <value>

  :CURSor:XY:CAX?

- **Functional Description**

  Set or query the horizontal position of cursor line A in the XY coordinate system.

  <value>: Horizontal position, where the range determined by the vertical range and vertical offset of the corresponding channel's X-axis.

- **Return Format**

  The query returns the horizontal position of cursor line A in scientific notation.

- **For Example**

| | |
|---|---|
| :CURSor:XY:CAX 0.1 | Set the horizontal position of cursor A to 100 mV. |
| :CURSor:XY:CAX? | The query returns 1.000000e-01. |

### :CURSor:XY:CBX

- **Command Format**

  :CURSor:XY:CBX <value>

  :CURSor:XY:CBX?

- **Functional Description**

  Set or query the horizontal position of cursor line B in the XY coordinate system.

  <value>: Horizontal position, where the range determined by the vertical range and vertical offset of the corresponding channel's X-axis.

- **Return Format**

  The query returns the horizontal position of cursor line B in scientific notation.

- **For Example**

| | |
|---|---|
| :CURSor:XY:CBX 0.1 | Set the horizontal position of cursor B to 100 mV. |
| :CURSor:XY:CBX? | The query returns 1.000000e-01. |

### :CURSor:XY:CAY

- **Command Format**

  :CURSor:XY:CAY <value>

  :CURSor:XY:CAY?

- **Functional Description**

  Set or query the vertical position of cursor line A in the XY coordinate system.

  <value>: Horizontal position, where the range determined by the vertical range and vertical offset of the corresponding channel's Y-axis.

- **Return Format**

  The query returns the vertical position of cursor line A in scientific notation. The unit is determined by the vertical unit of the selected channel's Y-axis.

- **For Example**

| | |
|---|---|
| :CURSor:XY:CAY 0.3 | Set the vertical position of cursor A to 300 mV. |
| :CURSor:XY:CAY? | The query returns 3.000000e-01. |

### :CURSor:XY:CBY

- **Command Format**

  :CURSor:XY:CBY <value>

  :CURSor:XY:CBY?

- **Functional Description**

  Set or query the vertical position of cursor line B in the XY coordinate system.

  <value>: Horizontal position, where the range determined by the vertical range and vertical offset of the corresponding channel's Y-axis.

- **Return Format**

The query returns the vertical position of cursor line B in scientific notation. The unit is determined by the vertical unit of the selected channel's Y-axis.

- **For Example**

| | |
|---|---|
| :CURSor:XY:CBY 0.3 | Set the vertical position of cursor B to 300 mV. |
| :CURSor:XY:CBY? | The query returns 3.000000e-01. |

### :CURSor:XY:AXValue?

- **Command Format**

:CURSor:XY:AXValue?

- **Functional Description**

Query the X value at cursor line A. The unit is determined by the vertical unit of the corresponding channel's X-axis.

In time measurement mode, the voltage at the intersection of cursor line A and the waveform of the corresponding channel in the YT coordinate system.

In voltage or screen measurement mode, the X value at cursor line A in the XY coordinate system.

- **Return Format**

The query returns the X value at cursor line A in scientific notation.

- **For Example**

| | |
|---|---|
| :CURSor:XY:AXV? | The query returns the X value at cursor A as 2.000000e-02. |

### :CURSor:XY:BXValue?

- **Command Format**

:CURSor:XY:BXValue?

- **Functional Description**

Query the X value at cursor B. The unit is determined by the vertical unit of the corresponding channel's X-axis.

In time measurement mode, the voltage at the intersection of cursor line B and the waveform of the corresponding channel in the YT coordinate system.

In voltage or screen measurement mode, the X value at cursor line B in the XY coordinate system.

- **Return Format**

The query returns the X value at cursor B in scientific notation.

- **For Example**

:CURSor:XY:BXV? The query returns the X value at cursor B as 2.000000e-02.

### :CURSor:XY:AYValue?

- **Command Format**

  :CURSor:XY:AYValue?

- **Functional Description**

  Query the Y value at cursor A. The unit is determined by the vertical unit of the corresponding channel's Y-axis.

  In time measurement mode, the voltage at the intersection of cursor line A and the waveform of the corresponding channel in the YT coordinate system.

  In voltage or screen measurement mode, the X value at cursor line B in the XY coordinate system.

- **Return Format**

  The query returns the Y value at cursor A in scientific notation.

- **For Example**

  :CURSor:XY:AYV? The query returns the Y value at cursor line A as 3.000000e-01.

### :CURSor:XY:BYValue?

- **Command Format**

  :CURSor:XY:BYValue?

- **Functional Description**

  Query the Y value at cursor B. The unit is determined by the vertical unit of the corresponding channel's Y-axis.

  In time measurement mode, the voltage at the intersection of cursor line B and the waveform of the corresponding channel in the YT coordinate system.

  In voltage or screen measurement mode, the X value at cursor line B in the XY coordinate system.

- **Return Format**

  The query returns the Y value at Cursor line B in scientific notation.

- **For Example**

  :CURSor:XY:BYV? The query returns the Y value at cursor B as 3.000000e-01.

### :CURSor:XY:XDELta?

- **Command Format**

:CURSor:XY:XDELta?

- **Functional Description**

Query the difference "ΔX" between cursor A and cursor B. The unit is determined by the vertical unit of the corresponding channel's X-axis.

In time measurement mode, the voltage difference between the intersection of cursor line A and cursor B and the waveform of the corresponding channel in the YT coordinate system.

In voltage or screen measurement mode, the difference between the X values of cursor line A and cursor B in the XY coordinate system.

- **Return Format**

The query returns the difference "ΔX" between cursor A and cursor B in scientific notation.

- **For Example**

:CURSor:XY:XDEL? The query returns the difference "ΔX" as 2.000000e-02.

### :CURSor:XY:YDELta?

- **Command Format**

:CURSor:XY:YDELta?

- **Functional Description**

Query the difference "ΔY" between cursor A and cursor B. The unit is determined by the vertical unit of the corresponding channel's Y-axis.

In time measurement mode, the voltage difference between the intersection of cursor line A and cursor B and the waveform of the corresponding channel in the YT coordinate system.

In voltage or screen measurement mode, the difference between the Y values of cursor line A and cursor B in the XY coordinate system.

- **Return Format**

The query returns the difference "ΔY" between cursor A and cursor B in scientific notation.

- **For Example**

:CURSor:XY:YDEL? The query returns the difference "ΔY" as 2.000000e-01.

### :CURSor:XY:A:RADius?

- **Command Format**

:CURSor:XY:A:RADius?

- **Functional Description**

Query the radius of the polar coordinate system formed by the voltage values of source X and source Y at cursor A.

- **Return Format**

  The query returns the radius of the polar coordinate system in scientific notation, with the unit in V.

- **For Example**

  :CURSor:XY:A:RADius? The query returns the radius of the polar coordinate system as 2.000000e+00.

### :CURSor:XY:B:RADius?

- **Command Format**

  :CURSor:XY:B:RADius?

- **Functional Description**

  Query the radius of the polar coordinate system formed by the voltage values of source X and source Y at cursor B.

- **Return Format**

  The query returns the radius of the polar coordinate system in scientific notation, with the unit in V.

- **For Example**

  :CURSor:XY:B:RADius? The query returns the radius of the polar coordinate system as 2.000000e+00.

### :CURSor:XY:DELTa:RADius?

- **Command Format**

  :CURSor:XY:DELTa:RADius?

- **Functional Description**

  Query the radius of the polar coordinate system formed by the voltage difference values of source X and source Y at cursor A and cursor B, respectively.

- **Return Format**

  The query returns the radius of the polar coordinate system in scientific notation, with the unit in V.

- **For Example**

  :CURSor:XY:DELTa:RADius? The query returns the radius of the polar coordinate system as 2.000000e+00.

### :CURSor:XY:A:ANGLe?

- **Command Format**

  :CURSor:XY:A:ANGLe?

- **Functional Description**

  Query the angle of the polar coordinate system formed by the voltage values of source X and source Y at cursor A.

- **Return Format**

  The query returns the angle of the polar coordinate system in scientific notation, with the unit in degree (°).

- **For Example**

  :CURSor:XY:A:ANGLe? The query returns the angle of the polar coordinate system as 2.000000e+00.

### :CURSor:XY:B:ANGLe?

- **Command Format**

  :CURSor:XY:B:ANGLe?

- **Functional Description**

  Query the angle of the polar coordinate system formed by the voltage values of source X and source Y at cursor B.

- **Return Format**

  The query returns the angle of the polar coordinate system in scientific notation, with the unit in degree (°).

- **For Example**

  :CURSor:XY:B:ANGLe? The query returns the angle of the polar coordinate system as 2.000000e+00.

### :CURSor:XY:DELTa:ANGLe?

- **Command Format**

  :CURSor:XY:DELTa:ANGLe?

- **Functional Description**

  Query the angle of the polar coordinate system formed by the voltage difference values of source X and source Y at cursor A and cursor B, respectively.

- **Return Format**

  The query returns the angle of the polar coordinate system in scientific notation, with the unit

in degree (°).

- **For Example**

:CURSor:XY:DELTa:ANGLe? The query returns the angle of the polar coordinate system as 2.000000e+00.

### :CURSor:XY:A:PRODuct?

- **Command Format**

:CURSor:XY:A:PRODuct?

- **Functional Description**

Query the product of the voltage values of source X and source Y at cursor A.

- **Return Format**

The query returns the product in scientific notation, with the unit in VV.

- **For Example**

:CURSor:XY:A:PRODuct? The query returns the product as 2.000000e+00.

### :CURSor:XY:B:PRODuct?

- **Command Format**

:CURSor:XY:B:PRODuct?

- **Functional Description**

Query the product of the voltage values of source X and source Y at cursor B.

- **Return Format**

The query returns the product in scientific notation, with the unit in VV.

- **For Example**

:CURSor:XY:B:PRODuct? The query returns the product as 2.000000e+00.

### :CURSor:XY:DELta:PRODuct?

- **Command Format**

:CURSor:XY:DELTa:PRODuct?

- **Functional Description**

Query the product of the voltage values of source X and source Y at cursor A and cursor B, respectivly.

- **Return Format**

The query returns the product in scientific notation, with the unit in VV.

- **For Example**

| | |
|---|---|
| :CURSor:XY:DELTa:PRODuct? | The query returns the product as 2.000000e+00. |

### :CURSor:XY:A:RATio?

- **Command Format**

  :CURSor:XY:A:RATio?

- **Functional Description**

  Query the ratio of the voltage values of source Y and source X at cursor A.

- **Return Format**

  The query returns the ratio in scientific notation, with the unit in VV.

- **For Example**

| | |
|---|---|
| :CURSor:XY:A:RATio? | The query returns the ratio as 2.000000e+00. |

### :CURSor:XY:B:RATio?

- **Command Format**

  :CURSor:XY:B:RATio?

- **Functional Description**

  Query the ratio of the voltage values of source Y and source X at cursor B.

- **Return Format**

  The query returns the ratio in scientific notation, with the unit in VV.

- **For Example**

| | |
|---|---|
| :CURSor:XY:B:RATio? | The query returns the ratio as 2.000000e+00. |

### :CURSor:XY:DELTa:RATio?

- **Command Format**

  :CURSor:XY:DELTa:RATio?

- **Functional Description**

  Query the ratio of the voltage values of source Y and source X at cursor A and cursor B, respectivly.

- **Return Format**

  The query returns the ratio in scientific notation, with the unit in VV.

- **For Example**

| | |
|---|---|
| :CURSor:XY:DELTa:RATio? | The query returns the ratio as 2.000000e+00. |

### :CURSor:XY?

- **Command Format**

  :CURSor:XY?

- **Functional Description**

  The query returns all measurement parameters at once in XY mode. The returned data is in CSV and conforms to the Data Block Format.

- **Return Format**

  The query returns all measurement parameters in scientific notation, using standard units.

- **For Example**

  :CURSor:XY? The query returns #9000000100

  NAME,A,B,DELTA

  t,2.000000e-02,2.000000e-02,2.000000e-02

  x,1.000000e-02,1.000000e-02,1.000000e-02

  y,1.000000e-02,1.000000e-02,1.000000e-02

  radius,1.000000e-02,1.000000e-02,1.000000e-02

  angle,1.000000e-02,1.000000e-02,1.000000e-02

  product,1.000000e-02,1.000000e-02,1.000000e-02

  ratio,1.000000e-02,1.000000e-02,1.000000e-02

## FILE Command

This command is used Set the reference waveforms and save functions.

### :FILE:LOAD

- **Command Format**

  :FILE:LOAD <filename>[,<source>][,<disk>]

- **Functional Description**

  Load the waveforms into the reference channel or Load setting data into the oscilloscope. <filename> represents the filename, which must be a character string enclosed in double quotation marks, for example, “test.bsv."

  A file name with a *.dat extension represents waveform data to be loaded into the reference channel, matching the oscilloscope’s suffix name.

  A file name with a *.set extension represents setting data to be loaded into the oscilloscope, matching the oscilloscope’s suffix name.

  A file name with a *.bode.csv extension represents Bode plot setting data to be loaded into the

oscilloscope, matching the oscilloscope's suffix name. Currently, it only works on a USB flash drive.

<source > represents the reference channel {REFA | REFB | REFC | REFD}. This parameter can be selected and is only available when loading the waveform data.

REFA represents the reference channel A.

REFB represents the reference channel B.

REFC represents the reference channel C.

REFD represents the reference channel D.

<disk> represents the memory medium { FLASH | UDISK }. This parameter can be selected. If this parameter is omitted, it indicates internal data stored in FLASH.

FLASH represents internal data stored in FLASH.

UDISK represents internal data stored in a USB flash drive.

- **For Example**

FILE:LOAD "test.dat",REFA,UDISK

Load the waveform data of a file test.dat on a USB flash drive to reference channel A.

FILE:LOAD "system-set-up01.set"

Load configuration data from position 1 on the internal storage medium to the oscilloscope.

FILE:LOAD "001.bode.csv"

Load the Bode plot from a USB flash drive to the oscilloscope.

Notes: When the oscilloscope cannot automatically define the file name and suffix, the file names for internal storage settings must be in the format "system-set-up01.set" and "system-set-up255.set", with a maximum limit of 255 files. Similarly, the file names for internal .bsv waveform files must be in the format "wave01.bsv" and "wave255.bsv", also with a maximum of 255 files.

## :FILE:SAVe

- **Command Format**

:FILE:SAVe <filename>[,<source>][,<disk>]

- **Functional Description**

Save the channel's waveform or setting data to a file.

<filename> represents the filename, which must be a character string enclosed in double quotation marks, for example, "test.bsv."

A file name with a *.dat or *.csv extension represents a channel's waveform to be saved into the file, matching the oscilloscope's suffix name.

A file name with a *.set extension represents setting data to be saved into the file, matching the

oscilloscope's suffix name.

A file name with a *.bode.csv extension represents Bode plot setting data to be saved into the file, matching the oscilloscope's suffix name. Currently, it only works on a USB flash drive.

A file name with a *.bmp, *.jpeg, or *.png represents a picture to be saved into the file, matching the oscilloscope's suffix name.

<source > represents the physical channel or the logic channel {CHANnel<n>|MATH<n>}. This parameter can be selected and is only available when loading the waveform data.

CHANnel<n> represents CHANne1, CHANne2, CHANne3, and CHANne4.

MATH<n> represents MATH1, MATH2, MATH3, and MATH4.

<disk> represents the memory medium { FLASH | UDISK }. This parameter can be selected. If this parameter is omitted, it indicates internal data stored in FLASH.

FLASH represents internal data stored in FLASH.

UDISK represents internal data stored in a USB flash drive.

- **For Example**

FILE:SAVe "test.dat",CHANnel1,UDISK

Save waveform data from channel 1 in the format test.dat to a USB flash drive.

FILE:SAVe "system-set-up01.set"

Save the oscilloscope's configuration data to position 1 in internal memory.

FILE:SAVe "wave01.dat",CHANnel1,FLASH

Save waveform data from channel 1 to a USB flash drive.

FILE:SAVe "wave01.dat",CHANnel1

Save waveform data from channel 1 to internal memory.

FILE:SAVe "system-set-up01.set",FLASH

Save the oscilloscope's configuration data to a USB flash drive.

FILE:SAVe "system-set-up01.set"

Save the oscilloscope's configuration data to position 1 in internal memory.

FILE:SAVe "001.bode.csv",UDISK

Save the Bode plot to the file “001.bode.csv” on a USB flash drive.

FILE:SAVe "test.png", UDISK

Save the picture data in the format ."png” to the file “001.bode.csv” on a USB flash drive.

Notes: When the oscilloscope cannot automatically define the file name and suffix, the file names for internal storage settings must be in the format “system-set-up01.set” and “system-set-up255.set”, with a maximum limit of 255 files. Similarly, the file names for internal .bsv waveform files must be in the format “wave01.bsv” and “wave255.bsv”, also with

a maximum of 255 files.

# RECord Command

This command is used Set the recording waveform functions of the oscilloscope.

### :RECord:ENABle

- **Command Format**

  :RECord:ENABle { {1|ON} | {0|OFF} }

  :RECord:ENABle?

- **Functional Description**

  Switch the waveform recording to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :RECord:ENABle ON Enable waveform recording.

  :RECord:ENABle? The query returns 1, indicating that the waveform recording is enabled.

### :RECord:FAST

- **Command Format**

  :RECord:FAST { {1|ON} | {0|OFF} }

  :RECord:FAST?

- **Functional Description**

  Switch the sequence acquisition (fast recording) to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :RECord:FAST ON Enable quick waveform recording.

  :RECord:FAST? The query returns 1, indicating that the quick waveform recording is enabled.

### :RECord:STARt

- **Command Format**

  :RECord:STARt { {1|ON} | {0|OFF} }

  :RECord:STARt?

- **Functional Description**

  Start (ON) or stop (OFF) the waveform recording.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :RECord:STARt ON | Start waveform recording. |
| :RECord:STARt? | The query returns 1, indicating that waveform recording is started. |

### :RECord:INTerval

- **Command Format**

  :RECord:INTerval <time>

  :RECord:INTerval?

- **Functional Description**

  Set the time interval for waveform recording.

- **Return Format**

  The query returns the time interval of waveform recording in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :RECord:INTerval 200ns | Set the time interval to 200 ns. |
| :RECord:INTerval? | The query returns 2.000000e-04. |

### :RECord:FRAMes

- **Command Format**

  :RECord:FRAMes <value>

  :RECord:FRAMes?

- **Functional Description**

  Set or query the frame for waveform recording.

- **Return Format**

  The query returns the waveform recording frame as an integer.

- **For Example**

| | |
|---|---|
| :RECord:FRAMes 400 | Set the waveform recording frame as 400. |
| :RECord:FRAMes? | The query returns 400. |

### :RECord:PROMpt

- **Command Format**

  :RECord:PROMpt { {1|ON} | {0|OFF} }

  :RECord:PROMpt?

- **Functional Description**

  Switch the beep of waveform recording to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :RECord:PROMpt ON | Enable the beep of waveform recording. |
| :RECord:PROMpt? | The query returns 1, indicating that the beep of waveform recording is enabled. |

### :RECord:PLAY

- **Command Format**

  :RECord:PLAY { {1|ON} | {0|OFF} }

  :RECord:PLAY?

- **Functional Description**

  Switch the playback of waveform recording to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :RECord:PLAY ON | Enable the playback of waveform recording. |
| :RECord:PLAY? | The query returns 1, indicating that the playback of waveform recording is enabled. |

### :RECord:PLAY:STARt

- **Command Format**

  :RECord:PLAY:STARt <value>

  :RECord:PLAY:STARt?

- **Functional Description**

  Set or query the start frame of playback for waveform recording.

- **Return Format**

  The query returns the start frame of playback for waveform recording as an integer.

- **For Example**

| | |
|---|---|
| :RECord:PLAY:STARt 10 | Set the start frame of playback for waveform recording to 10. |
| :RECord:PLAY:STARt? | The query returns 10. |

### :RECord:PLAY:STOP

- **Command Format**

:RECord:PLAY:STOP <value>

:RECord:PLAY:STOP?

- **Functional Description**

Set or query the end frame of playback for waveform recording.

- **Return Format**

The query returns the end frame of playback for waveform recording as an integer.

- **For Example**

| | |
|---|---|
| :RECord:PLAY:STOP 100 | Set the end frame of playback for waveform recording to 100. |
| :RECord:PLAY:STOP? | The query returns 100. |

### :RECord:PLAY:MODe

- **Command Format**

:RECord:PLAY:MODe {LOOP|SINGle}

:RECord:PLAY:MODe?

- **Functional Description**

Set or query the mode for waveform recording.

LOOP: Loop playback

SINGle: Single playback

- **Return Format**

The query returns {LOOP|SINGle}.

- **For Example**

| | |
|---|---|
| :RECord:PLAY:MODe LOOP | Set the playback mode to LOOP. |
| :RECord:PLAY:MODe? | The query returns LOOP. |

### :RECord:PLAY:DIRection

- **Command Format**

:RECord:PLAY:DIRection {FORWard|BACKward}

:RECord:PLAY:DIRection?

- **Functional Description**

  Set or query the playback direction for waveform recording.

  FORWard: forward play

  BACKward: backward play

- **Return Format**

  The query returns {FORWard|BACKward}.

- **For Example**

| | |
|---|---|
| :RECord:PLAY:DIRection FORWard | Set the playback direction to FORWard. |
| :RECord:PLAY:DIRection? | The query returns FORWard. |

### :RECord:PLAY:CURRent

- **Command Format**

  :RECord:PLAY:CURRent <value>

  :RECord:PLAY:CURRent?

- **Functional Description**

  Set or query the current playback frame for waveform recording.

- **Return Format**

  The query returns the current playback frame as an integer.

- **For Example**

| | |
|---|---|
| :RECord:PLAY:CURRent 100 | Set the current playback frame to 100. |
| :RECord:PLAY:CURRent? | The query returns 100. |

### :RECord:PLAY:INTerval

- **Command Format**

  :RECord:PLAY:INTerval <time>

  :RECord:PLAY:INTerval?

- **Functional Description**

  Set the playback interval for waveform recording.

- **Return Format**

  The query returns the playback interval in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :RECord:PLAY:INTerval 20ms | Set the playback interval to 20 ms. |
| :RECord:PLAY:INTerval? | The query returns 2.000000e-02. |

### :RECord:PLAY:FIRSt

- **Command Format**

  :RECord:PLAY:FIRSt

- **Functional Description**

  Set the playback to the first frame of waveform recording.

- **For Example**

  :RECord:PLAY:FIRSt Set the playback to the first frame of waveform recording.

### :RECord:PLAY:LAST

- **Command Format**

  :RECord:PLAY:LAST

- **Functional Description**

  Set the playback to the last frame of waveform recording.

- **For Example**

  :RECord:PLAY:LAST Set the playback to the last frame of waveform recording.

### :RECord:PLAY:NEXT

- **Command Format**

  :RECord:PLAY:NEXT

- **Functional Description**

  Set the playback to the next frame of waveform recording.

- **For Example**

  :RECord:PLAY:NEXT Set the playback to the next frame of waveform recording.

### :RECord:PLAY:BACK

- **Command Format**

  :RECord:PLAY:BACK

- **Functional Description**

  Set the playback to the previous frame of waveform recording.

- **For Example**

  :RECord:PLAY:BACK Set the playback to the previous frame of waveform recording.

## DVM Command

This command is used Set the digital voltmeter function of the oscilloscope.

### :DVM:ENABle

- **Command Format**

  :DVM:ENABle { {1|ON} | {0|OFF} }

  :DVM:ENABle?

- **Functional Description**

  Set or query the state of the digital voltmeter function: ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :DVM:ENABle ON | Enable the digital voltmeter. |
| :DVM:ENABle? | The query returns 1. |

### :DVM:SOURce

- **Command Format**

  :DVM:SOURce <source>

  :DVM:SOURce?

- **Functional Description**

  Set or query the source for the digital voltmeter.

  <source>: CHANnel<n>, where n can take values 1, 2, 3, or 4.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4}.

- **For Example**

| | |
|---|---|
| :DVM:SOURce CHANnel1 | Set the source as Channel 1. |
| :DVM:SOURce? | The query returns CHANnel1. |

### :DVM:MODe

- **Command Format**

  :DVM:MODe {ACRMs|DC|DCRMs}

  :DVM:MODe?

- **Functional Description**

  Set or query the mode for the digital voltmeter.

  ACRMs: Displays the RMS (Root Mean Square) value of all sampled data with removed DC components.

  DC: Displays the average value of all sampled data.

DCRMs: Displays the RMS (Root Mean Square) value of all sampled data.

- **Return Format**

The query returns {ACRMs|DC|DCRMs}.

- **For Example**

| | |
|---|---|
| :DVM:MODe DC | Set the mode for the digital voltmeter to DC. |
| :DVM:MODe? | The query returns DC. |

### :DVM:INTerval

- **Command Format**

:DVM:INTerval <time>

:DVM:INTerval?

- **Functional Description**

Set the refresh interval for the digital voltmeter.

- **Return Format**

The query returns the current time interval, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :DVM:INTerval 1 | Set the refresh interval time for the digital voltmeter to 1s. |
| :DVM:INTerval? | The query returns 1.000000e+00. |

### :DVM:BEEP?

- **Command Format**

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

:DVM:BEEP?

- **Functional Description**

Set and query the beep state for the digital voltmeter.

- **Return Format**

The query returns the beep state: 0 indicates OFF, while 1 indicates ON.

- **For Example**

| | |
|---|---|
| :DVM:BEEP ON | Enable the beep of the digital voltmeter. |
| :DVM:BEEP? | The query returns 1, indicating that the beep of the digital voltmeter is enabled. |

### :DVM:QUALifier

- **Command Format**

:DVM:QUALifier { GREaterthan | LESSthan | INRange | OUTRange }

:DVM:QUALifier?

- **Functional Description**

Set the alarm limit condition for the digital voltmeter to GREaterthan (Greater than), LESSthan (Less than), INRange (Within the range), or OUTRange (Outside of the range).

- **Return Format**

The query returns { GREaterthan | LESSthan | INRange | OUTRange }.

- **For Example**

| | |
|---|---|
| :DVM:QUALifier GRE | Set the slope condition to GREaterthan. |
| :DVM:QUALifier? | The query returns GREaterthan. |

### :DVM:UPPer

- **Command Format**

DVM:UPPer <upper>

DVM:UPPer?

- **Functional Description**

Set the upper limit of the voltage for the digital voltmeter.

- **Return Format**

The query returns the current upper limit of the voltage, with the unit in V.

- **For Example**

| | |
|---|---|
| DVM:UPPer 1 | Set the upper limit of the voltage for the digital voltmeter to 1 V. |
| DVM:UPPer? | The query returns 1.000000e+00. |

### :DVM:LOWer

- **Command Format**

:DVM:LOWer <lower>

:DVM:LOWer?

- **Functional Description**

Set the lower limit of the voltage for the digital voltmeter.

- **Return Format**

The query returns the current lower limit of the voltage, with the unit in V.

- **For Example**

| | |
|---|---|
| :DVM:LOWer -1 | Set the lower limit of the voltage for the digital voltmeter to -1 V. |
| :DVM:LOWer? | The query returns -1.000000e+00. |

### :DVM:CURRent?

- **Command Format**

  :DVM:CURRent?

- **Functional Description**

  Query the currently measured voltage with the digital voltmeter.

- **Return Format**

  The query returns the currently measured voltage with the digital voltmeter in scientific notation, using standard units. The unit is determined by the command :DVM:MODe.

- **For Example**

  :DVM:CURRent? The currently measured voltage is 3.000000e-03.

### :DVM:STATistic?

- **Command Format**

  :DVM:STATistic? <type>

- **Functional Description**

  Obtain the statistical results of the digital voltmeter, and the measurement-related settings are also effective for this function.

  <type> represents statistical type: {MAXimum|MINimum|CURRent|AVERages|DEViation} represents maximum, minimum, current value, average, and variance, respectively.

- **Return Format**

  The query returns the statistical results with standard units in scientific notation.

- **For Example**

  :POWer:QUALity:STATistic:ITEM? MAX,VRMS

  The query returns the maximum statistical value of RMS voltage 1.120000e+00 under digital voltmeter.

### :DVM:STATistic:HISTogram:RESult?

- **Command Format**

  :DVM:STATistic:HISTogram:RESult?

- **Functional Description**

  Obtain the histogram statistical results of the digital voltmeter, sorted by the left boundary value, right boundary value, and probability percentage. The measurement-related settings also apply to this function.

- **Return Format**

The query returns the histogram statistical results arranged in CSV format in scientific notation. The returned data conforms to the Data Block Format.

- **For Example**

:DVM:STATistic:HISTogram:RESult?

The query returns the histogram statistical results of the digital voltmeter:

#9000000128HISTOGRAM,

Sum,Peaks,Max,Min,Pk_Pk,Mean,Median,Mode,Width,Siqma

100, 93, -8.000mV, -16.000mV,8.000mV,-8.400mV,-8.000mV,-8.000mV, 2.400mV

In which, “#9000000148” is the TMC data block header, followed by the data in the option list. In the data block header, the number following “#9” indicates the number of bytes of valid data that follows. HISTogram represents the histogram, with each piece of data separated by commas and each line of data separated by newline characters.

The statistical results include the following items.

Sum: Total count of statistical data.

Peaks: The maximum count of data being counted.

Max: The maximum value of the total statistical data.

Min: The minimum value of the total statistical data.

Pk_Pk: The difference between the maximum and minimum values (Max-Min) in the total statistical data.

Mean: The average of histogram.

Median: The median value of histogram.

Mode: The mode value of histogram.

Sigma: The standard deviation of histogram.

# COUNter Command

This command is used Set the frequency meter of the oscilloscope.

### :COUNter:ENABle

- **Command Format**

:COUNter:ENABle { {1|ON} | {0|OFF} }

:COUNter:ENABle?

- **Functional Description**

Set or query the state of the frequency meter: ON or OFF.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :COUNter:ENABle ON | Enable the frequency meter. |
| :COUNter:ENABle? | The query returns 1. |

### :COUNter:SOURce

- **Command Format**

:COUNter:SOURce <source>

:COUNter:SOURce?

- **Functional Description**

Set or query the source for the frequency meter.

<source>: {CHANnel<n> | TRIGger}.

CHANnel<n>: Physical channel {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

TRIGger: Trigger source includes the external source and digital channel.

- **Return Format**

The query returns {CHANnel<n> | TRIGger}.

- **For Example**

| | |
|---|---|
| :COUNter:SOURce CHANnel1 | Set the source as Channel 1. |
| :COUNter:SOURce? | The query returns CHANnel1. |

### :COUNter:MODe

- **Command Format**

:COUNter:MODe {FREQuency|PERiod|TOTalize}

:COUNter:MODe?

- **Functional Description**

Set or query the mode for the frequency meter.

FREQuency: Frequency measurement

PERiod: Period measurement

TOTalize: Accumulation measurement

- **Return Format**

The query returns {FREQuency|PERiod|TOTalize}.

- **For Example**

| | |
|---|---|
| :COUNter:MODe FREQuency | Set the mode for the frequency meter to FREQuency. |
| :COUNter:MODe? | The query returns FREQuency. |

### :COUNter:INTerval

- **Command Format**

  :COUNter:INTerval <time>

  :COUNter:INTerval?

- **Functional Description**

  Set or query the refresh interval for the frequency meter.

- **Return Format**

  The query returns the refresh interval of the frequency meter in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :COUNter:INTerval 500ms | Set the refresh interval to 500 ms. |
| :COUNter:INTerval? | The query returns 5.000000e-01. |

### :COUNter:EFFective:DIGits

- **Command Format**

  :COUNter:EFFective:DIGits <val>

  :COUNter:EFFective:DIGits?

- **Functional Description**

  Set or query the effective digits for the frequency measurement.

  <val>: Effective digit, represented as an integer, ranging from 3 to 7.

- **Return Format**

  The query returns the effective digit as an integer.

- **For Example**

| | |
|---|---|
| :COUNter:EFFective:DIGits 3 | Set the effective digit to 3. |
| :COUNter:EFFective:DIGits? | The query returns 3. |

### :COUNter:CURRent?

- **Command Format**

  :COUNter:CURRent?

- **Functional Description**

  Query the currently measured value for the frequency meter.

- **Return Format**

  The query returns the currently measured value of the frequency meter in scientific notation, using standard units. The unit is determined by the command :COUNter:MODe.

- **For Example**

:COUNter:CURRent? The currently measured value is 3.000000e+003.

### :COUNter:STATistic?

- **Command Format**

:COUNter:STATistic? <type>

- **Functional Description**

Obtain the statistical results of the frequency meter, and the measurement-related settings are also effective for this function.

<type> represents statistical type: {MAXimum|MINimum|CURRent|AVERages|DEViation} represents maximum, minimum, current value, average, and variance, respectively.

- **Return Format**

The query returns the statistical results with standard units in scientific notation.

- **For Example**

:POWer:QUALity:STATistic:ITEM? MAX,VRMS

The query returns the maximum statistical value of RMS voltage 1.000000e+03 under frequency voltmeter.

### :COUNter:STATistic:HISTogram:RESult?

- **Command Format**

:COUNter:STATistic:HISTogram:RESult?

- **Functional Description**

Obtain the histogram statistical results of the frequency meter, sorted by the left boundary value, right boundary value, and probability percentage. The measurement-related settings also apply to this function.

- **Return Format**

The query returns the histogram statistical results arranged in CSV format in scientific notation. The returned data conforms to the Data Block Format.

- **For Example**

:COUNter:STATistic:HISTogram:RESult?

The query returns the histogram statistical results of the frequency meter:

#9000000128HISTOGRAM,

Sum,Peaks,Max,Min,Pk_Pk,Mean,Median,Mode, Siqma

100,93,8.000Hz,16.000Hz,8.000Hz,8.000Hz,8.000Hz,8.000Hz,2.000Hz

In which, “#9000000148” is the TMC data block header, followed by the data in the option list. In the data block header, the number following “#9” indicates the number of bytes of valid data that follows. HISTogram represents the histogram, with each piece of data separated by commas and each line of data separated by newline characters.

The statistical results include the following items.

Sum: Total count of statistical data.

Peaks: The maximum count of data being counted.

Max: The maximum value of the total statistical data.

Min: The minimum value of the total statistical data.

Pk_Pk: The difference between the maximum and minimum values (Max-Min) in the total statistical data.

Mean: The average of histogram.

Median: The median value of histogram.

Mode: The mode value of histogram.

Sigma: The standard deviation of histogram.

### :COUNter:CLEar

- **Command Format**

  :COUNter:CLEar

- **Functional Description**

  Clear the total count.

- **For Example**

| | |
|---|---|
| :COUNter:CLEar | Clear the total count. |

## PF Command

This command is used Set the Pass/Fail test function of the oscilloscope.

### :PF:ENABle

- **Command Format**

  :PF:ENABle { {1|ON} | {0|OFF} }

  :PF:ENABle?

- **Functional Description**

  Set or query the state of the Pass/Fail test function: ON or OFF.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :PF:ENABle ON | Enable the Pass/Fail test function. |
| :PF:ENABle? | The query returns 1. |

### :PF:SOURce

- **Command Format**

:PF:SOURce <source>

:PF:SOURce?

- **Functional Description**

Set or query the measurement source for the Pass/Fail test.

<source>: CHANnel<n>, where n can take values 1, 2, 3, or 4.

- **Return Format**

The query returns {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4}.

- **For Example**

| | |
|---|---|
| :PF:SOURce CHANnel1 | Set the measurement source as Channel 1. |
| :PF:SOURce? | The query returns CHANnel1. |

### :PF:OPERate

- **Command Format**

:PF:OPERate {RUN|STOP}

:PF:OPERate?

- **Functional Description**

Start or stop the Pass/Fail test.

- **Return Format**

The query returns {RUN|STOP}.

- **For Example**

| | |
|---|---|
| :PF:OPERate RUN | Start the Pass/Fail test. |
| :PF:OPERate? | The query returns RUN. |

### :PF:RESet

- **Command Format**

:PF:RESet

- **Functional Description**

Reset the frames, failed frames, and total frames that have passed through the Pass/Fail test.

- **For Example**

| | |
|---|---|
| :PF:RESet | Reset the results of the Pass/Fail test. |

### :PF:OUTPut:ENABle

- **Command Format**

:PF:OUTPut:ENABle { {1|ON} | {0|OFF} }

:PF:OUTPut:ENABle?

- **Functional Description**

Set or query the output state of AUX OUT port on the rear panel: ON or OFF.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :PF:OUTPut:ENABle ON | Enable AUX OUT. |
| :PF:OUTPut:ENABle? | The query returns 1. |

### :PF:OUTPut:QUALifier

- **Command Format**

:PF:OUTPut:QUALifier {PASS|FAILED}

:PF:OUTPut:QUALifier?

- **Functional Description**

Set or query the output condition for the Pass/Fail test.

- **Return Format**

The query returns {PASS|FAILED}.

- **For Example**

| | |
|---|---|
| :PF:OUTPut:QUALifier PASS | Set the output condition for the Pass/Fail test to PASS. |
| :PF:OUTPut:QUALifier? | The query returns PASS. |

### :PF:OUTPut:POLarity

- **Command Format**

:PF:OUTPut:POLarity {POSitive | NEGative}

:PF:OUTPut:POLarity?

- **Functional Description**

Set or query the output polarity for the Pass/Fail test to POSitive (Positive pulse) or NEGative

(Negative pulse).

- **Return Format**

The query returns { POSitive | NEGative }.

- **For Example**

| | |
|---|---|
| :PF:OUTPut:POLarity POSitive | Set the output polarity to POSitive. |
| :PF:OUTPut:POLarity? | The query returns POSitive. |

### :PF:OUTPut:TIMe

- **Command Format**

:PF:OUTPut:TIMe <time>

:PF:OUTPut:TIMe?

- **Functional Description**

Set or query the output pulse time for the Pass/Fail test.

<time>: Pulse time, with the unit in seconds (s).

- **Return Format**

The query returns the pulse time in scientific notation, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :PF:OUTPut:TIMe 2us | Set the output pulse time for the Pass/Fail test to 2 us. |
| :PF:OUTPut:TIMe? | The query returns 2.000000e-06. |

### :PF:STOP:TYPe

- **Command Format**

:PF:STOP:TYPe {PCOUNT|FCOUNT}

:PF:STOP:TYPe?

- **Functional Description**

Set or query the stop type for the Pass/Fail test.

PCOUNT represents the pass count; FCOUNT represents the failed count.

- **Return Format**

The query returns {PCOUNT|FCOUNT}.

- **For Example**

| | |
|---|---|
| :PF:STOP:TYPe PCOUNT | Set the stop type for the Pass/Fail test to PCOUNT. |
| :PF:STOP:TYPe? | The query returns PCOUNT. |

### :PF:STOP:QUALifier

- **Command Format**

  :PF:STOP:QUALifier {LEQual | GEQual}

  :PF:STOP:QUALifier?

- **Functional Description**

  Set or query the stop condition for the Pass/Fail test.

  GEQual represents greater than or equal to

  LEQual represents less than or equal to

- **Return Format**

  The query returns {LEQual | GEQual}.

- **For Example**

| | |
|---|---|
| :PF:STOP:QUALifier GEQual | Set the stop condition for the Pass/Fail test to GEQual (≥). |
| :PF:STOP:QUALifier? | The query returns GEQual. |

### :PF:STOP:THReshold

- **Command Format**

  :PF:STOP:THReshold <value>

  :PF:STOP:THReshold?

- **Functional Description**

  Set or query the stop threshold for the stop condition in the Pass/Fail test.

  <value>: Stop threshold, ranging from 1-10000. The range is automatically adjusted to suit different oscilloscopes.

- **Return Format**

  The query returns the stop threshold as an integer.

- **For Example**

| | |
|---|---|
| :PF:STOP:THReshold 100 | Set the stop threshold for the Pass/Fail test to 100. |
| :PF:STOP:THReshold? | The query returns 100. |

### :PF:SCReen

- **Command Format**

  :PF:SCReen { {1|ON} | {0|OFF} }

  :PF:SCReen?

- **Functional Description**

  Set or query whether the auto-save screenshot function is ON or OFF during Pass/Fail test.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :PF:SCReen ON | Enable auto-save screenshot function during Pass/Fail test. |
| :PF:SCReen? | The query returns 1. |

### :PF:TEMPlate:SOURce

- **Command Format**

  :PF:TEMPlate:SOURce {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | REF | USB}

  :PF:TEMPlate:SOURce?

- **Functional Description**

  Set or query the template source for the Pass/Fail test.

  The physical channel {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4} can be used as the template source.

  If the template source is set to REF, use the command :PF:TEMPlate:LOAD Load the waveform file form REF as the template source. If the template source is set to USB, use the command :PF:TEMPlate:LOAD Load the waveform file form a USB flash drive as the template source.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | REF | USB}.

- **For Example**

| | |
|---|---|
| :PF:TEMPlate:SOURce CHANnel1 | Set the template source as Channel 1. |
| :PF:TEMPlate:SOURce? | The query returns CHANnel1. |

### :PF:TEMPlate:CREate

- **Command Format**

  :PF:TEMPlate:CREate

- **Functional Description**

  Set the current horizontal and vertical adjustments for creating the Pass/Fail test rules.

- **For Example**

| | |
|---|---|
| :PF:TEMPlate:CREate | Create the Pass/Fail test rules. |

### :PF:TEMPlate:LOAD

- **Command Format**

:PF:TEMPlate:LOAD <filepath>

- **Functional Description**

Load the specified rule file as the template source.

<filepath> represents the absolute file path of the file, which must be enclosed in double quotation marks as string data.

Note: In the path string, Local:/ refers to the internal instrument storage, while Udisk:/ refers to the external USB flash drive.

- **For Example**

| | |
|---|---|
| :PF:TEMPlate:LOAD "Local:/wave/test.tmp" | Load the specified rule file test.tmp from local storage. |
| :PF:TEMPlate:LOAD "Udisk:/wave/test.tmp" | Load the specified rule file test.tmp from USB flash drive. |

### :PF:TEMPlate:SAVe

- **Command Format**

:PF:TEMPlate:SAVe <filepath>

- **Functional Description**

Save the current horizontal and vertical adjustments for creating the Pass/Fail test rules to a file.

<filepath> represents the absolute file path of the file, which must be enclosed in double quotation marks as string data.

Note: In the path string, Local:/ refers to the internal instrument storage, while Udisk:/ refers to the external USB flash drive.

- **For Example**

| | |
|---|---|
| :PF:TEMPlate:SAVe "Local:/wave/test.tmp" | Save the rule to the test.tmp file in local storage. |
| :PF:TEMPlate:SAVe "Udisk:/wave/test.tmp" | Save the rule to the test.tmp file in USB flash drive. |

### :PF:TEMPlate:X

- **Command Format**

:PF:TEMPlate:X <value>

:PF:TEMPlate:X?

- **Functional Description**

Set or query the horizontal tolerance set by the Pass/Fail test template.

<value>: Horizontal tolerance, ranging from 1 to 100. The range is automatically adjusted to suit the different oscilloscopes.

- **Return Format**

The query returns the horizontal tolerance set by the template as an integer.

- **For Example**

| | |
|---|---|
| :PF:TEMPlate:X 50 | Set the horizontal tolerance set by the template to 50. |
| :PF:TEMPlate:X? | The query returns 50. |

### :PF:TEMPlate:Y

- **Command Format**

:PF:TEMPlate:Y <value>

:PF:TEMPlate:Y?

- **Functional Description**

Set or query the vertical tolerance set by the Pass/Fail test template.

<value>: Vertical tolerance, ranging from 1 to 100. The range is automatically adjusted to suit the different oscilloscopes.

- **Return Format**

The query returns the vertical tolerance set by the template as an integer.

- **For Example**

| | |
|---|---|
| :PF:TEMPlate:Y 50 | Set the vertical tolerance set by the template to 50. |
| :PF:TEMPlate:Y? | The query returns 50. |

### :PF:RESult?

- **Command Format**

:PF:RESult?

- **Functional Description**

Query the statistical results for the Pass/Fail test.

Returned data format: <pass>, <failed>, and <total>, where <pass> represents the pass count, <failed> represents the failed count, and <total> represents the total count.

- **Return Format**

The query returns the statistical results of the Pass/Fail test.

- **For Example**

| | |
|---|---|
| :PF:RESult? | The query returns 35, 42, and 77. |

# ACQuire Command

This command is used Set the acquisition mode for the oscilloscope.

### :ACQuire:TYPe

- **Command Format**

  :ACQuire:TYPe {NORMal | AVERage | PEAKdetect | HRESolution | ENHanced }

  :ACQuire:TYPe?

- **Functional Description**

  Set the acquisition mode for the oscilloscope to NORMal (Normal), AVERage (Average), PEAKdetect (Peak), HRESolution (High resolution), or | ENHanced.

- **Return Format**

  The query returns {NORMal | AVERage | PEAKdetect | HRESolution | ENHanced }.

- **For Example**

| | |
|---|---|
| :ACQ:TYPe AVER | Set the acquisition mode to AVERage. |
| :ACQ:TYPe? | The query returns AVERage. |

### :ACQuire:AVERages:COUNt

- **Command Format**

  :ACQuire:AVERages:COUNt <count>

  :ACQuire:AVERages:COUNt?

- **Functional Description**

  Set the average count in the oscilloscope's averaging sampling mode, where <count> steps in powers of 2, ranging from 2 to 8192, with 1 ≤ N ≤ 30.

- **Return Format**

  The query returns the current average count in average mode.

- **For Example**

| | |
|---|---|
| :ACQ:AVER:COUN 32 | Set the average count in the oscilloscope's averaging sampling mode to 32. |
| :ACQ:AVER:COUN? | The query returns 32. |

### :ACQuire:MEMory:DEPTh

- **Command Format**

  :ACQuire:MEMory:DEPTh { AUTO | 25K | 250K | 500K | 5M | 50M | 100M}

  :ACQuire:MEMory:DEPTh?

- **Functional Description**

  Set the storage depth mode, which is automatically adjusted to suit different oscilloscopes.

- **Return Format**

  The query returns { AUTO | 25K | 250K | 500K | 5M | 50M | 100M}.

- **For Example**

| | |
|---|---|
| :ACQ:MEM:DEPT AUTO | Set the storage depth mode to AUTO. |
| :ACQ:MEM:DEPT? | The query returns AUTO. |

### :ACQuire:INTerpolation:MODe

- **Command Format**

  :ACQuire:INTerpolation:MODe { LINear | SINC}

  :ACQuire:INTerpolation:MODe?

- **Functional Description**

  Set the interpolation mode for the acquisition mode.

  LINear: Linear interpolation

  SINC: Whittaker-Shannon interpolation

- **Return Format**

  The query returns { LINear | SINC}.

- **For Example**

| | |
|---|---|
| :ACQuire:INTerpolation:MODe LINear | Set the interpolation mode to LINear. |
| :ACQuire:INTerpolation:MODe? | The query returns LINear. |

### :ACQuire:ENHanced:RESolution

- **Command Format**

  :ACQuire:ENHanced:RESolution { OFF | 1 | 1.5 | 2 | 2.5 | 3 | 4 }

  :ACQuire:ENHanced:RESolution?

- **Functional Description**

  Set the enhanced resolution function.

- **Return Format**

  The query returns { OFF | 1 | 1.5 | 2 | 2.5 | 3 | 4 }.

- **For Example**

| | |
|---|---|
| :ACQuire:ENHanced:RESolution OFF | Disable the enhanced resolution. |
| :ACQuire:ENHanced:RESolution? | The query returns OFF. |

# DISPlay Command

This command is used Set or query the display function or data of the oscilloscope.

### :DISPlay:DATA?

- **Command Format**

  :DISPlay:DATA? [<format>]

- **Functional Description**

  Query the picture data in the corresponding picture format on the oscilloscope's current screen.

  <format>: Picture data in three formats {BMP|PNG|JPG}. If the parameter is omitted, BMP is the default format.

- **Return Format**

  The query returns the picture data in the corresponding picture format. The returned data conforms to the Data Block Format.

- **For Example**

| | |
|---|---|
| :DISPlay:DATA? | The query returns the picture data in BMP format. |
| :DISPlay:DATA? PNG | The query returns the picture data in PNG format. |

### :DISPlay:FORMat

- **Command Format**

  :DISPlay:FORMat { VECTors | DOTS }

  :DISPlay:FORMat?

- **Functional Description**

  Set the display format for the sampling point to VECTors (Vectors) or DOTS (Dots).

- **Return Format**

  The query returns { VECTors | DOTS }.

- **For Example**

| | |
|---|---|
| :DISPlay:FORMat VECT | Set the display format for the sampling point to VECTors. |
| :DISPlay:FORMat | The query returns VECTors. |

### :DISPlay:GRID

- **Command Format**

  :DISPlay:GRID {FULL|HALF|CROSS|NONE}

  :DISPlay:GRID?

- **Functional Description**

  Set the display format of the grid.

- **Return Format**

  The query returns {FULL|HALF|CROSS|NONE}.

- **For Example**

| | |
|---|---|
| :DISPlay:GRID CROSS | Set the display format of the grid to CROSS to hide the divide grids. |
| :DISPlay:GRID? | The query returns CROSS. |

### :DISPlay:GRAD:TIMe

- **Command Format**

  :DISPlay:GRAD:TIMe {AUTO|50ms|100ms|200ms|500ms|1s|2s|5s|10s|20s|INFinite|OFF}

  :DISPlay:GRAD:TIMe?

- **Functional Description**

  Set the persistence time.

- **Return Format**

  The query returns {AUTO|50ms|100ms|200ms|500ms|1s|2s|5s|10s|20s|INFinite|OFF}.

- **For Example**

| | |
|---|---|
| :DISPlay:GRAD:TIMe 50ms | Set the persistence time to 50 ms. |
| :DISPlay:GRAD:TIMe? | The query returns 50 ms. |

### :DISPlay:STOP:GRAD

- **Command Format**

  :DISPlay:STOP:GRAD { {1|ON} | {0|OFF} }

  :DISPlay:STOP:GRAD?

- **Functional Description**

  Set or query the persistence display in stop state.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :DISPlay:STOP:GRAD ON | Enable the persistence display in stop state. |
| :DISPlay:STOP:GRAD? | The query returns 1, indicating that the persistence display is enabled in stop state. |

### :DISPlay:GRID:BRIGhtness

- **Command Format**

  :DISPlay:GRID:BRIGhtness <count>

  :DISPlay:GRID:BRIGhtness?

- **Functional Description**

  Set the grid brightness, where <count> can take values from 0 to 100, the larger the number, the brighter the grid.

- **Return Format**

  The query returns the current grid brightness.

- **For Example**

  :DISPlay:GRID:BRIGhtness 50 Set the grid brightness to 50%.

  :DISPlay:GRID:BRIGhtness? The query returns 50.

### :DISPlay:WAVe:BRIGhtness

- **Command Format**

  :DISPlay:WAVe:BRIGhtness <count>

  :DISPlay:WAVe:BRIGhtness?

- **Functional Description**

  Set the waveform brightness, where <count> can take values from 1 to 100, the larger the number, the brighter the waveform.

- **Return Format**

  The query returns the current waveform brightness.

- **For Example**

  :DISPlay:WAVe:BRIGhtness 50 Set the waveform brightness to 50%.

  :DISPlay:WAVe:BRIGhtness? The query returns 50.

### :DISPlay:BACKlight:BRIGhtness

- **Command Format**

  :DISPlay:BACKlight:BRIGhtness <count>

  :DISPlay:BACKlight:BRIGhtness?

- **Functional Description**

  Set the screen backlight brightness, where <count> can take values from 1 to 100, the larger the number, the brighter the screen.

- **Return Format**

The query returns the screen backlight brightness.

- **For Example**

| | |
|---|---|
| :DISPlay:BACKlight:BRIGhtness 50 | Set the screen backlight brightness to 50%. |
| :DISPlay:BACKlight:BRIGhtness? | The query returns 50. |

### :DISPlay:WINDow:TRANsparent

- **Command Format**

:DISPlay:WINDow:TRANsparent <count>

:DISPlay:WINDow:TRANsparent?

- **Functional Description**

Set the transparency of UI window, where <count> can take values from 0 to 100, the larger the number, the brighter the screen.

- **Return Format**

The query returns the transparency of UI window.

- **For Example**

| | |
|---|---|
| :DISPlay:WINDow:TRANsparent 50 | Set the transparency of UI window to 50%. |
| :DISPlay:WINDow:TRANsparent? | The query returns 50. |

### :DISPlay:COLOR

- **Command Format**

:DISPlay:COLOR { {1|ON} | {0|OFF} }

:DISPlay:COLOR?

- **Functional Description**

Switch the color temperature to ON or OFF.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :DISPlay:COLOR ON | Enable the color temperature. |
| :DISPlay:COLOR? | The query returns 1, indicating that the color temperature is enabled. |

### :DISPlay:CLEar

- **Command Format**

:DISPlay:CLEar

- **Functional Description**

Clear and refresh the waveform on the oscilloscope's screen. If there is a reference waveform, it will also be cleared and refreshed.

# WAVeform Command

This command is used Read the waveform data and the related parameters on the oscilloscope's screen.

### :WAVeform:MODe

- **Command Format**

  :WAVeform:MODe {NORMal | RAW}

  :WAVeform:MODe?

- **Functional Description**

  Set or query the read mode for waveform data. Changing the mode will reset the start point, stop point, and waveform point parameters.

  NORMal: Read the current waveform data on the screen, this waveform data point is fixed.

  RAW: Read the waveform data from the internal storage, the waveform data point is related to the storage depth. The data in the internal storage can only be read when the oscilloscope in the stop state.

- **Return Format**

  The query returns {NORMal | RAW}.

- **For Example**

  :WAVeform:MODe RAW Set the read mode for waveform data to RAW.

  :WAVeform:MODe? The query returns RAW.

### :WAVeform:FORMat

- **Command Format**

  :WAVeform:FORMat {WORD | DWORD | ASCII }

  :WAVeform:FORMat?

- **Functional Description**

  Set or query the return format of waveform data.

  WORD: It is only available in the analog channel, returning the AD value as an integer in two bytes for each waveform point.

  DWORD: It is only available in the analog channel and math channels. For the analog channel, it returns the AD value as a real-type in four bytes for each waveform point. For the digital

channel, it returns the high and low level state data of all digital channel signals in four bytes, with each digital channel occupying 1 bit, arranged according to the channel serial number from smallest to largest.

ASCII : It is only available in the analog channel and MATH channels. It returns the actual voltage value of each waveform point in scientific notation, with each voltage value separated by comma and conforming to the Data Block Format.

For example,: #90000012342.00000e+01,2.20000e+01, 2.30000e+01.....\n.

- **Return Format**

The query returns { WORD | DWORD | ASCII }.

- **For Example**

| | |
|---|---|
| :WAVeform:FORMat DWORD | Set the return format for waveform AD data to DWORD. |
| :WAVeform:FORMat? | The query returns DWORD. |

### :WAVeform:STARt

- **Command Format**

:WAVeform:STARt <start>

:WAVeform:STARt?

- **Functional Description**

Set or query the start position for reading waveform data.

<start>: The start position for reading waveform data.

When the waveform mode is data displayed on the screen, the reading range is from 1 to the maximum number of waveform points currently displayed on the screen.

When the waveform mode is waveform data in internal storage, the reading range is from 1 to the current maximum storage depth.

- **Return Format**

The query returns the start position as an integer.

- **For Example**

| | |
|---|---|
| :WAVeform:STARt 700 | Set the start position for reading waveform data to 700. |
| :WAVeform:STARt? | The query returns 700. |

### :WAVeform:STOP

- **Command Format**

:WAVeform:STOP <stop>

:WAVeform:STOP?

- **Functional Description**

Set or query the stop position for reading waveform data.

<stop>: The stop position for reading waveform data.

When the waveform mode is data displayed on the screen, the reading range is from 1 to the number of waveform points currently displayed on the screen. The default value is the waveform point currently displayed on the screen.

When the waveform mode is waveform data in internal storage, the reading range is from 1 to the current maximum storage depth. The default value is the currently saved storage point.

- **Return Format**

The query returns the stop position as an integer.

- **For Example**

| | |
|---|---|
| :WAVeform:STOP 1400 | Set the stop position for reading waveform data to 1, 400. |
| :WAVeform:STOP? | The query returns 1,400. |

### :WAVeform:POINts

- **Command Format**

:WAVeform:POINts <points>

:WAVeform:POINts?

- **Functional Description**

Set or query the waveform point to be read.

When the waveform mode is data displayed on the screen, it returns the default waveform point on the screen if no configured.

When the waveform mode is waveform data in internal storage, it returns the waveform point of the current storage depth by default if no configured.

- **Return Format**

The query returns the waveform point to be read.

- **For Example**

| | |
|---|---|
| :WAVeform:POINts 1400 | Set the waveform point to be read to 1,400. |
| :WAVeform:POINts? | The query returns 1,400. |

### :WAVeform:SOURce

- **Command Format**

:WAVeform:SOURce {CHANnel<n>|MATH<n>|DIGital}

:WAVeform:SOURce?

- **Functional Description**

Set or query the source of waveform data to be queried. If this command is not set, it represents querying the waveform data of the current channel. Setting waveform data source will reset the start point, stop point, and waveform point parameters.

CHANnel<n>: Physical channel {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

MATH<n>: Logic channel {MATH1|MATH2|MATH3|MATH4}.

DIGital: Digital channel represents uses D0-D15 logic channels and analog channels as the source for digital channels.

When the channel source is set to MATH channel, only ASCII format data is valid, it returns the waveform voltage data displayed on the screen for the math channel.

When the channel source is set to a digital channel, only DWORD format data is valid. It returns the high and low level status data for all digital channels. Each digital channel accounts for 1 bit, arranged according to the channel serial number from smallest to largest. All digital channels together account for four bytes.

- **Return Format**

The query returns

{ CHANnel1|CHANnel2|CHANnel3|CHANnel4|MATH1|MATH2|MATH3|MATH4|DIGital}.

- **For Example**

| | |
|---|---|
| :WAVeform:SOURce CHAN1 | Set the source of waveform data to be queried to Channel 1. |
| :WAVeform:SOURce? | The query returns CHANnel1. |

### :WAVeform:DATA?

- **Command Format**

:WAVeform:DATA?

- **Functional Description**

Read the waveform data of the specified channel.

The waveform data source is determined by the command :WAVeform:SOURce, and the data format is specified by the command :WAVeform:FORMat.

Note:

When the channel source is set to MATH channel, only ASCII format data is valid, it returns the waveform voltage data displayed on the screen for the math channel.

When the channel source is set to a digital channel, only DWORD format data is valid. It returns the high and low level status data for all digital channels. Each digital channel accounts for 1 bit, arranged according to the channel serial number from smallest to largest. All digital channels

together account for four bytes.

- **Return Format**

The query returns the waveform data, with the format specified by the command :WAVeform:FORMat, conforming to the Data Block Format.

- **For Example**

The process for obtaining waveform data from the specified analog channel is as follows.

The sequence for obtaining waveform data displayed on the screen:

:WAVeform:SOURce CHAN1 Set the signal source for querying waveform data to channel 1.

:WAVeform:MODE NORMal Set Read the waveform data displayed on the screen.

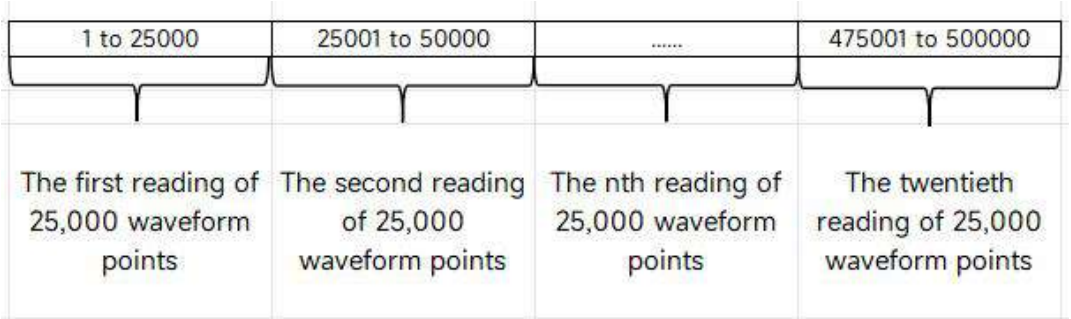
:WAVeform:FORMat WORD Set the return format for waveform data to WORD.

:WAVeform:DATA? Obtain the waveform data.

The sequence for obtaining waveform data saved in the internal memory:

The process of obtaining waveform data from internal storage involves reading block by block if the storage depth is too large. This process is only valid when the oscilloscope is in the stop state.

For example, reading a waveform with a storage depth of 500K points: the maximum number of points that can be read at one time is 25,000. Therefore, it is necessary Read a minimum of 20 times (500,000 / 25,000).




:WAVeform:SOURce CHAN1 Set the signal source for querying waveform data to channel 1.

:WAVeform:MODE RAW Set Read the waveform data from internal storage.

:WAVeform:FORMat WORD Set the return format for waveform data to WORD.

:WAVeform:POINts 25000 Set the reading waveform point form internal storage as 25,000.

Obtain the complete internal data, repeatedly send the read waveform command.

:WAVeform:DATA? Obtain a block of waveform data from internal data.

:WAVeform:START? Continue reading waveform data until the start position is equal to -1, indicating that the last point has been reached.

Explanation:

When reading internal data in batches, each read data comes from a specific area in the internal storage, and the waveform data between two adjacent pieces is continuous. Each piece of data conforms to the Data Block Format.

Data Conversion Formula:

AD value converts to voltage:

voltage = [ (data value - yreference) * yincrement ] + yorigin

in here, data value represents AD value; yreference represents the vertical reference in the Y direction; yincrement represents the voltage value of each ADC unit in the Y direction; yorigin represents the vertical offset in the Y direction relative to the vertical reference position.

Time Conversion Formula:

time = [ (data point number - xreference) * xincrement ] + xorigin

in here, data point number represents the serial number of waveform points; xreference represents the time reference of waveform points in the X direction; xincrement represents the time interval between adjacent two points in the X direction; xorigin represents the start time interval of waveform points in the X direction.

### :WAVeform:XINCrement?

- **Command Format**

  :WAVeform:XINCrement?

- **Functional Description**

  Query the time interval between two adjacent points in the X direction for the currently selected channel.

  When the waveform mode is data displayed on the screen, XINCrement=TimeScale/125.

  When the waveform mode is waveform data in internal storage, XINCrement=1/SampleRate.

- **Return Format**

  The query returns the time base, with the unit in seconds (s).

- **For Example**

  :WAV:XINC? The query returns 3.000000e-03.

### :WAVeform:XORigin?

- **Command Format**

  :WAVeform:XORigin?

- **Functional Description**

  Query the start time of waveform data in the X direction for the currently selected channel.

When the waveform mode is data displayed on the screen, XORigin = TimeScale*4.

When the waveform mode is waveform data in internal storage, XORigin = (SamplePoints/SampleRate)/2.

- **Return Format**

The query returns the time, with the unit in seconds (s).

- **For Example**

:WAV:XOR? The query returns 3.000000e-02.

### :WAVeform:XREFerence?

- **Command Format**

:WAVeform:XREFerence?

- **Functional Description**

Query the time reference of waveform point in the X direction for the currently selected channel.

- **Return Format**

The query returns the time reference, which is 0.

- **For Example**

:WAV:XREF? The query returns 0.

### :WAVeform:YINCrement?

- **Command Format**

:WAVeform:YINCrement?

- **Functional Description**

Query the ADC unit of voltage in the Y direction for the currently selected channel. The unit conforms to the current amplitude unit.

YINCrement = VerticalScale/25.

- **Return Format**

The query returns the unit of voltage in the Y direction.

- **For Example**

:WAV:YINC? The query returns 2.000000e+00.

### :WAVeform:YORigin?

- **Command Format**

:WAVeform:YORigin?

- **Functional Description**

  Query the vertical offset relative to the vertical reference position in the Y direction for the currently selected channel.

  YORigin = VerticalOffset/YINCrement.

- **Return Format**

  The query returns the vertical offset as an integer.

- **For Example**

  :WAV:YOR? The query returns 0.

### :WAVeform:YREFerence?

- **Command Format**

  :WAVeform:YREFerence?

- **Functional Description**

  Query the vertical reference position and ADC of channel's zero level in the Y direction for the currently selected channel.

- **Return Format**

  The query returns the reference position as an integer.

- **For Example**

  :WAV:YREF? The query returns 128.

### :WAVeform:PREamble?

- **Command Format**

  :WAVeform:PREamble?

- **Functional Description**

  The query returns the waveform parameters of the current system in the following sequence: Format, Type, Points, Count, Xinc, Xor, Xref, Yinc, Yor, and Yref.

  Format: Waveform format (WORD, BYTe, and ASCII ).

  Type: Acquisition mode NORMal, AVERage, PEAKdetec, HRESolution, and ENHanced.

  Points: Waveform data point to be read.

  Count: In average sampling mode, this represents the average time; in other modes, it is 1.

  Xinc: The time difference between two points in the X direction of the waveform data source.

  Xor: The relative time of the trigger point.

  Xref: X reference

  Yinc: The unit of voltage in Y direction.

Yor: The Y direction relative to zero position of YREF.

Yref: The reference value and ADC Y of channel's zero level in the Y direction.

- **Return Format**

The query returns waveform parameters where integer data is represented as integers and real numeric data is represented in scientific notation. The returned data conforms to the Data Block Format.

- **For Example**

:WAVeform:PREamble?

The query returns "#9000001000ASCII, NORMal, 1400, 1, 8.000e-009, -6.000e-006, 0, 4.000e-002, 0.000e000, 128."

# BUS Command

This command is used Set the bus decoding for RS232, SPI, $I^2C$, CAN, CANFD, USB, LIN, FlexRay, AUDio, Manchester, SENT, and Arinc429 on the oscilloscope. It has the ability to decode four groups.

## Essential Attribute

### :BUS<n>:DISPlay

- **Command Format**

:BUS<n>:DISPlay { {1|ON} | {0|OFF} }

:BUS<n>:DISPlay?

- **Functional Description**

Switch the bus decoding of the oscilloscope to ON or OFF.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :BUS1:DISPlay ON | Enable the bus decoding and display the decoded waveform. |
| :BUS1:DISPlay? | The query returns 1. |

### :BUS<n>:TYPe

- **Command Format**

:BUS<n>:TYPe{RS232|I2C|SPI|CAN|CANFD|CANXL|LIN|FR|AUDio||M1553|MANC|SENT|A429|ONEWire|I3C}:BUS<n>:TYPe?

- **Functional Description**

Select the bus protocol type on the oscilloscope.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

RS232 (UART/RS232), I2C (I2C bus), SPI (SPI bus), CAN (CAN bus), CANFD (CAN-FD bus), CANXL (CANXL bus),LIN (LIN bus), FR (FlexRay bus), AUDio (AUDIO bus), M1553 (MIL - STD - 1553B bus), MANC (Manchester), SENT (SENT bus), A429 (Arinc429 bus), ONEWire (1-wire bus), and I3C (I3C bus).

- **Return Format**

The query returns {RS232|I2C|SPI|CAN|CANFD|CANXL|LIN|FR|AUDIO|M1553|MANC|SENT|A429|ONEWire|I3C}.

- **For Example**

| | |
|---|---|
| :BUS1:TYPe I2C | Select the bus protocol type to I2C. |
| :BUS1:TYPe? | The query returns I2C. |

### :BUS<n>:FORMat

- **Command Format**

:BUS<n>:FORMat {ASCII | BINary | HEX | DEC}

:BUS<n>:FORMat?

- **Functional Description**

Set the bus format on the oscilloscope.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns {ASCII | BINary | HEX | DEC}.

- **For Example**

| | |
|---|---|
| :BUS1:FORMat BIN | Set the bus format to BINary. |
| :BUS1:FORMat? | The query returns BINary. |

### :BUS<n>:EVENt

- **Command Format**

:BUS<n>:EVENt { {1|ON} | {0|OFF} }

:BUS<n>:EVENt?

- **Functional Description**

Switch the bus decoding event of the oscilloscope to ON or OFF.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :BUS1:EVENt ON | Enable the bus decoding event. |
| :BUS1:EVENt? | The query returns 1. |

### :BUS<n>:DATA?

- **Command Format**

  :BUS<n>:DATA?

- **Functional Description**

  Read the data from the decoding event table on the oscilloscope.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

  The query returns the data from the decoding event table. The returned data conforms to the Data Block Format.

- **For Example**

  :BUS1:DATA? The query returns:

  #9000000089RS232,

  TIME,DATA,CHECK,

  -1us,0,0,

  -890.5ns,1,0,

  -403.4ns,0,0,

  9.8ns,1,0,

  531.7ns,0,0,

  RS232 represents a decoding type (which may also be I2C, SPI, CAN, etc.), followed immediately by the event table data in CSV format. The specified format of the event table data is automatically adapted by different devices. The data are separated by commas and will automatically line wrap according to the decoding list. The data value is related to the system display settings.

### :BUS<n>:POSition

- **Command Format**

  :BUS<n>:POSition <value>

  :BUS<n>:POSition?

- **Functional Description**

Set the position of bus decoding line on the oscilloscope.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<value>: Bus decoding line position, stepping by 6, ranging from -160 to 160. The center of the screen is considered zero, with positive values above and negative values below.

- **Return Format**

The query returns the vertical position value as an integer.

- **For Example**

| | |
|---|---|
| :BUS1:POSition 10 | Set the vertical position value for the bus to 10. |
| :BUS1:POSition? | The query returns 10. |

### :BUS<n>:LABel:ENABle

- **Command Format**

:BUS<n>:LABel:ENABle { {1|ON} | {0|OFF} }

:BUS<n>:LABel:ENABle?

- **Functional Description**

Set or query the enabling state of the specified decoding label.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :BUS1:LABel:ENABle ON | Enable the label of decoding 1 the label of decoding 1. |
| :BUS1:LABel:ENABle? | The query returns 1, indicating that the label of decoding 1 is enabled. |

## RS232

### :BUS<n>:RS232:SOURce

- **Command Format**

:BUS<n>:RS232:SOURce <source>

:BUS<n>:RS232:SOURce?

- **Functional Description**

Set or query the source of the decoding.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:SOURce CHANnel1 | Set the source as Channel 1. |
| :BUS1:RS232:SOURce? | The query returns CHANnel1. |

### :BUS<n>:RS232:LEVel

- **Command Format**

:BUS<n>:RS232:LEVel <level>

:BUS<n>:RS232:LEVel?

- **Functional Description**

Set or query the threshold.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Threshold value

- **Return Format**

The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:LEVel 2 | Set the level to 2 V. |
| :BUS1:RS232:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:RS232:POLarity

- **Command Format**

:BUS<n>:RS232:POLarity {POSitive|NEGative }

:BUS<n>:RS232:POLarity?

- **Functional Description**

Set or query the pulse polarity to POSitive (Positive) or NEGative (Negative).

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns the pulse polarity { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:POLarity POS | Set the pulse polarity to POSitive. |

| | |
|---|---|
| :BUS1:RS232:POLarity? | The query returns POSitive. |

### :BUS<n>:RS232:ORDer

- **Command Format**

  :BUS<n>:RS232:ORDer {LSB|MSB}

  :BUS<n>:RS232:ORDer?

- **Functional Description**

  Set or query the byte order for the RS232 bus.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  LSB: Least significant bit

  MSB: Most significant bit

- **Return Format**

  The query returns {LSB|MSB}.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:ORDer LSB | Set the byte order to LSB. |
| :BUS1:RS232:ORDer? | The query returns LSB. |

### :BUS<n>:RS232:BAUDrate

- **Command Format**

  :BUS<n>:RS232:BAUDrate <baud rate>

  :BUS<n>:RS232:BAUDrate?

- **Functional Description**

  Set or query the baud rate for the RS232 bus. The default unit is bps, and the parameter is an integer.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

  The query returns the baud rate.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:BAUDrate 9600 | Set the baud rate of the RS232 to 9600 bps. |
| :BUS1:RS232:BAUDrate? | The query returns 9,600. |

### :BUS<n>:RS232:WIDTh

- **Command Format**

  :BUS<n>:RS232:WIDTh {5|6|7|8}

:BUS<n>:RS232:WIDTh?

- **Functional Description**

Set or query the data bit width for the RS232 bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns {5|6|7|8}.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:WIDTh 6 | Set the data bit width for the RS232 bus to 6. |
| :BUS1:RS232:WIDTh? | The query returns 6. |

### :BUS<n>:RS232:STOP

- **Command Format**

:BUS<n>:RS232:STOP {1|2}

:BUS<n>:RS232:STOP?

- **Functional Description**

Set or query the stop bit for the RS232 bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns {1|2}.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:STOP 1 | Set the stop bit for the RS232 bus to 1. |
| :BUS1:RS232:STOP? | The query returns 1. |

### :BUS<n>:RS232:PARity

- **Command Format**

:BUS<n>:RS232:PARity {EVEN | ODD | NONE}

:BUS<n>:RS232:PARity?

- **Functional Description**

Set or query the parity check for the RS232 bus.

EVEN represents even check, ODD represents odd check, and NONE represents no check.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns {EVEN | ODD | NONE}.

- **For Example**

| | |
|---|---|
| :BUS1:RS232:PARity ODD | Set the parity check for the RS232 bus to ODD. |
| :BUS1:RS232:PARity? | The query returns ODD. |

## I2C

### :BUS<n>:I2C:SDA

- **Command Format**

  :BUS<n>:I2C:SDA <source>

  :BUS<n>:I2C:SDA?

- **Functional Description**

  Set or query the data source.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:I2C:SDA CHANnel1 | Set the data source as Channel 1. |
| :BUS1:I2C:SDA? | The query returns CHANnel1. |

### :BUS<n>:I2C:SCL

- **Command Format**

  :BUS<n>:I2C:SCL <source>

  :BUS<n>:I2C:SCL?

- **Functional Description**

  Set or query the clock source.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:I2C:SCL CHANnel1 | Set the clock source as Channel 1. |

:BUS1:I2C:SCL? The query returns CHANnel1.

### :BUS<n>:I2C:DLEVel

- **Command Format**

:BUS<n>:I2C:DLEVel <level>

:BUS<n>:I2C:DLEVel?

- **Functional Description**

Set or query the level value of the data line.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Level value

- **Return Format**

The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

:BUS1:I2C:DLEVel 2 Set the level value of the data line to 2 V.

:BUS1:I2C:DLEVel? The query returns 2.000000e+00.

### :BUS<n>:I2C:CLEVel

- **Command Format**

:BUS<n>:I2C:CLEVel <level>

:BUS<n>:I2C:CLEVel?

- **Functional Description**

Set or query the level value of the clock line.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Level value

- **Return Format**

The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

:BUS1:I2C:CLEVel 2 Set the level value of the clock line to 2 V.

:BUS1:I2C:CLEVel? The query returns 2.000000e+00.

### :BUS<n>:I2C:AWIDth

- **Command Format**

:BUS<n>:I2C:AWIDTh {7 | 10}

:BUS<n>:I2C:AWIDTh?

- **Functional Description**

Set or query the address bit width for the I2C bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns {7 | 10}.

- **For Example**

| | |
|---|---|
| :BUS1:I2C:AWIDTh 7 | Set the address bit width to 7. |
| :BUS1:I2C:AWIDTh? | The query returns 7. |

## SPI

### :BUS<n>:SPI:SCL

- **Command Format**

:BUS<n>:SPI:SCL <source>

:BUS<n>:SPI:SCL?

- **Functional Description**

Set or query the clock source.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:SCL CHANnel1 | Set the clock source as Channel 1. |
| :BUS1:SPI:SCL? | The query returns CHANnel1. |

### :BUS<n>:SPI:SMOSI

- **Command Format**

:BUS<n>:SPI:SMOSI <source>

:BUS<n>:SPI:SMOSI?

- **Functional Description**

Set or query the source of the MOSI data line.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:SMOSI CHANnel1 | Set the source of the MOSI data line to Channel 1. |
| :BUS1:SPI:SMOSI? | The query returns CHANnel1. |

### :BUS<n>:SPI:SCS

- **Command Format**

:BUS<n>:SPI:SCS <source>

:BUS<n>:SPI:SCS?

- **Functional Description**

Set or query the source of the chip select.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:SCS CHANnel1 | Set the source of the chip select to Channel 1. |
| :BUS1:SPI:SCS? | The query returns CHANnel1. |

### :BUS<n>:SPI:CLOCk:LEVel

- **Command Format**

:BUS<n>:SPI:CLOCk:LEVel <level>

:BUS<n>:SPI:CLOCk:LEVel?

- **Functional Description**

Set or query the level value of the clock line.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Level value

- **Return Format**

  The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:CLOCk:LEVel 2 | Set the level value of the clock line to 2 V. |
| :BUS1:SPI:CLOCk:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:SPI:MOSI:LEVel

- **Command Format**

  :BUS<n>:SPI:MOSI:LEVel <level>

  :BUS<n>:SPI:MOSI:LEVel?

- **Functional Description**

  Set or query the level value of the MOSI data line.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <level>: Level value

- **Return Format**

  The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:MOSI:LEVel 2 | Set the level value of the MOSI data line to 2 V. |
| :BUS1:SPI:MOSI:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:SPI:CS:LEVel

- **Command Format**

  :BUS<n>:SPI:CS:LEVel <level>

  :BUS<n>:SPI:CS:LEVel?

- **Functional Description**

  Set or query the level value of the chip select line.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <level>: Level value

- **Return Format**

  The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:CS:LEVel 2 | Set the level value of the chip select line to 2 V. |
| :BUS1:SPI:CS:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:SPI:CLOCk:POLarity

- **Command Format**

  :BUS<n>:SPI:CLOCk:POLarity {POSitive|NEGative}

  :BUS<n>:SPI:CLOCk:POLarity?

- **Functional Description**

  Set or query the polarity of the clock pulse to POSitive (Positive) or NEGative (Negative).

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

  The query returns the pulse polarity { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:CLOCk:POLarity POS | Set the pulse polarity of the clock line to POSitive. |
| :BUS1:SPI:CLOCk:POLarity? | The query returns POSitive. |

### :BUS<n>:SPI:MOSI:POLarity

- **Command Format**

  :BUS<n>:SPI:MOSI:POLarity {POSitive|NEGative }

  :BUS<n>:SPI:MOSI:POLarity?

- **Functional Description**

  Set or query the pulse polarity for the MOSI data line to POSitive (Positive) or NEGative (Negative).

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

  The query returns the pulse polarity { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:MOSI:POLarity POS | Set the pulse polarity of the MOSI data line to POSitive. |
| :BUS1:SPI:MOSI:POLarity? | The query returns POSitive. |

### :BUS<n>:SPI:CS:POLarity

- **Command Format**

  :BUS<n>:SPI:CS:POLarity {POSitive|NEGative }

  :BUS<n>:SPI:CS:POLarity?

- **Functional Description**

  Set or query the pulse polarity for the chip select line to POSitive (Positive) or NEGative (Negative).

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

  The query returns the pulse polarity { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:CS:POLarity POS | Set the pulse polarity of the chip select line to POSitive. |
| :BUS1:SPI:CS:POLarity? | The query returns POSitive. |

### :BUS<n>:SPI:DWIDth

- **Command Format**

  :BUS<n>:SPI:DWIDth <width>

  :BUS<n>:SPI:DWIDth?

- **Functional Description**

  Set or query the data bit width for the SPI bus.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <width>: Bit width, ranging from 4 to 32.

- **Return Format**

  The query returns the data bit width as an integer.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:DWIDth 4 | Set the data bit width to 4. |
| :BUS1:SPI:DWIDth? | The query returns 4. |

### :BUS<n>:SPI:ORDer

- **Command Format**

  :BUS<n>:SPI:ORDer {LSB|MSB}

  :BUS<n>:SPI:ORDer?

- **Functional Description**

  Set or query the byte order for the SPI bus.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  LSB: Least significant bit

  MSB: Most significant bit

- **Return Format**

The query returns {LSB|MSB}.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:ORDer LSB | Set the byte order to LSB. |
| :BUS1:SPI:ORDer? | The query returns LSB. |

### :BUS<n>:SPI:MODe

- **Command Format**

:BUS<n>:SPI:MODe { CS | TIMeout }

:BUS<n>:SPI:MODe?

- **Functional Description**

Set or query the decoding mode for the SPI bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns { CS | TIMeout }.

- **For Example**

| | |
|---|---|
| :BUS1:SPI:MODe TIMeout | Set the decoding mode to TIMeout. |
| :BUS1:SPI:MODe? | The query returns TIMeout. |

### :BUS<n>:SPI:TIMe

- **Command Format**

:BUS<n>:SPI:TIMe <time>

:BUS<n>:SPI:TIMe?

- **Functional Description**

Set or query the idle time in timeout mode for the SPI bus.

- **Return Format**

The query returns the current idle time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :BUS1:SPI:TIMe 1 | Set the idle time for the SPI bus to 1s. |
| :BUS1:SPI:TIMe? | The query returns 1.000000e+00. |

## CAN

### :BUS<n>:CAN:SOURce

- **Command Format**

:BUS<n>:CAN:SOURce <source>

:BUS<n>:CAN:SOURce?

- **Functional Description**

Set or query the source.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:CAN:SOURce CHANnel1 | Set the source as Channel 1. |
| :BUS1:CAN:SOURce? | The query returns CHANnel1. |

### :BUS<n>:CAN:LEVel

- **Command Format**

:BUS<n>:CAN:LEVel <level>

:BUS<n>:CAN:LEVel?

- **Functional Description**

Set or query the level value.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Level value

- **Return Format**

The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:CAN:LEVel 2 | Set the level to 2 V. |
| :BUS1:CAN:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:CAN:STYPe

- **Command Format**

:BUS<n>:CAN:STYPe { L | H }

:BUS<n>:CAN:STYPe?

- **Functional Description**

Set or query the signal type for the CAN bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

H: Actual CAN_H bus signal

L: Actual CAN_L bus signal

- **Return Format**

The query returns { L | H }.

- **For Example**

| | |
|---|---|
| :BUS1:CAN:STYPe H | Set the signal type to H. |
| :BUS1:CAN:STYPe? | The query returns H. |

### :BUS<n>:CAN:BAUDrate

- **Command Format**

:BUS<n>:CAN:BAUDrate <baud rate>

:BUS<n>:CAN:BAUDrate?

- **Functional Description**

Set or query the signal baud rate for the CAN bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<baud rate>: Baud rate, ranging from 10,000 to 1,000,000 bps.

- **Return Format**

The query returns the baud rate as an integer.

- **For Example**

| | |
|---|---|
| :BUS1:CAN:BAUDrate 100000 | Set the signal baud rate for the CAN bus to 100 kbps. |
| :BUS1:CAN:BAUDrate? | The query returns 100,000. |

## CAN-FD

### :BUS<n>:CANFD:SOURce

- **Command Format**

:BUS<n>:CANFD:SOURce <source>

:BUS<n>:CANFD:SOURce?

- **Functional Description**

Set or query the source.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

  :BUS1:CANFD:SOURce CHANnel1 Set the source as Channel 1.

  :BUS1:CANFD:SOURce? The query returns CHANnel1.

### :BUS<n>:CANFD:LEVel

- **Command Format**

  :BUS<n>:CANFD:LEVel <level>

  :BUS<n>:CANFD:LEVel?

- **Functional Description**

  Set or query the level value.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <level>: Level value

- **Return Format**

  The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

  :BUS1:CANFD:LEVel 2 Set the level to 2 V.

  :BUS1:CANFD:LEVel? The query returns 2.000000e+00.

### :BUS<n>:CANFD:STYPe

- **Command Format**

  :BUS<n>:CANFD:STYPe { L | H }

  :BUS<n>:CANFD:STYPe?

- **Functional Description**

  Set or query the signal type for the CAN-FD bus.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  H: Actual CAN_H bus signal

  L: Actual CAN_L bus signal

- **Return Format**

  The query returns { L | H }.

- **For Example**

  :BUS1:CANFD:STYPe H Set the signal type for the CAN-FD bus to H.

:BUS1:CANFD:STYPe? The query returns H.

### :BUS<n>:CANFD:BAUDrate

- **Command Format**

:BUS<n>:CANFD:BAUDrate <baud rate>

:BUS<n>:CANFD:BAUDrate?

- **Functional Description**

Set or query the signal baud rate for the CAN-FD bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<baud rate>: Baud rate, ranging from 10,000 to 1,000,000 bps.

- **Return Format**

The query returns the baud rate as an integer.

- **For Example**

:BUS1:CANFD:BAUDrate 100000 Set the signal baud rate for the CAN-FD bus to 100 kbps.

:BUS1:CANFD:BAUDrate? The query returns 100, 000.

### :BUS<n>:CANFD:FD:BAUDrate

- **Command Format**

:BUS<n>:CANFD:FD:BAUDrate <baud rate>

:BUS<n>:CANFD:FD:BAUDrate?

- **Functional Description**

Set or query the FD baud rate for the CAN-FD bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<baud rate>: Baud rate, ranging from 250,000 to 8,000,000 bps.

- **Return Format**

The query returns the baud rate as an integer.

- **For Example**

:BUS1:CANFD:FD:BAUDrate 500000 Set the FD baud rate for the CAN-FD bus to 500 kbps.

:BUS1:CANFD:FD:BAUDrate? The query returns 500, 000.

### :BUS<n>:CANFD:SPOSition

- **Command Format**

:BUS<n>:CANFD:SPOSition <position>

:BUS<n>:CANFD:SPOSition?

- **Functional Description**

  Set or query the sampling position for the CAN-FD bus signal.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <position>: Sampling position, ranging from 30-90, with the unit in %.

- **Return Format**

  The query returns the sampling position in scientific notation.

- **For Example**

| | |
|---|---|
| :BUS1:CANFD:SPOSition 65 | Set the sampling position of CAN-FD bus signal to 65%. |
| :BUS1:CANFD:SPOSition? | The query returns 6.500000e+01. |

## LIN

### :BUS<n>:LIN:SOURce

- **Command Format**

  :BUS<n>:LIN:SOURce <source>

  :BUS<n>:LIN:SOURce?

- **Functional Description**

  Set or query the source.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:LIN:SOURce CHANnel1 | Set the source as Channel 1. |
| :BUS1:LIN:SOURce? | The query returns CHANnel1. |

### :BUS<n>:LIN:LEVel

- **Command Format**

  :BUS<n>:LIN:LEVel <level>

  :BUS<n>:LIN:LEVel?

- **Functional Description**

  Set or query the level value.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Level value

- **Return Format**

The query returns the level value in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:LIN:LEVel 2 | Set the level to 2 V. |
| :BUS1:LIN:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:LIN:POLarity

- **Command Format**

:BUS<n>:LIN:POLarity {NORMal | INVert}

:BUS<n>:LIN:POLarity?

- **Functional Description**

Set the polarity for the LIN bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

NORMal (Normal, high=1)

INVert (Invert, high=0)

- **Return Format**

The query returns {NORMal | INVert}.

- **For Example**

| | |
|---|---|
| :BUS1:LIN:POLarity NORMal | Set the polarity to NORMal. |
| :BUS1:LIN:POLarity? | The query returns NORMal. |

### :BUS<n>:LIN:VERSion

- **Command Format**

:BUS<n>:LIN:VERSion {VER1|VER2|ANY}

:BUS<n>:LIN:VERSion?

- **Functional Description**

Set or query the version for the LIN bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

{VER1| VER2|ANY}: {VER1| VER2|ANY} represents V1.x, V2.x, and any arbitrary version, respectively.

- **Return Format**

The query returns {VER1|VER2|ANY}.

- **For Example**

:BUS1:LIN:VERSion VER1 Set the version to VER1.

:BUS1:LIN:VERSion? The query returns VER1.

### :BUS<n>:LIN:BAUDrate

- **Command Format**

:BUS<n>:LIN:BAUDrate <baud rate>

:BUS<n>:LIN:BAUDrate?

- **Functional Description**

Set or query the baud rate for the LIN bus signal.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<baud rate>: Baud rate, ranging from 1,000 to 1,000,000 bps.

- **Return Format**

The query returns the baud rate as an integer.

- **For Example**

:BUS1:LIN:BAUDrate 2400 Set the signal baud rate for the LIN bus to 2.4 kbps.

:BUS1:LIN:BAUDrate? The query returns 2, 400.

### :BUS<n>:LIN:ID:PARity

- **Command Format**

:BUS<n>:LIN:ID:PARity { {1|ON} | {0|OFF} }

:BUS<n>:LIN:ID:PARity?

- **Functional Description**

Set or query whether the ID includes parity bit for the LIN bus: ON indicates yes, while OFF indicates no.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

:BUS1:LIN:ID:PARity ON Enable parity bit for the ID.

:BUS1:LIN:ID:PARity? The query returns 1.

### :BUS<n>:LIN:LENGth:CTL

- **Command Format**

:BUS<n>:LIN:LENGth:CTL { {1|ON} | {0|OFF} }

:BUS<n>:LIN:LENGth:CTL?

- **Functional Description**

Set or query whether the data length should be set for the LIN bus: ON indicates yes, while OFF indicates no.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :BUS1:LIN:LENGth:CTL ON | Enable the data length. |
| :BUS1:LIN:LENGth:CTL? | The query returns 1. |

### :BUS<n>:LIN:LENGth

- **Command Format**

:BUS<n>:LIN:LENGth <length>

:BUS<n>:LIN:LENGth?

- **Functional Description**

Set or query the data length for the LIN trigger. When using this command, it is enabled by default.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<length>: Data length, ranging from 1-8.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :BUS1:LIN:LENGth 6 | Set the data length to 6. |
| :BUS1:LIN:LENGth? | The query returns 6. |

## AUDIO

### :BUS<n>:AUDio:FORMat

- **Command Format**

:BUS<n>:AUDio:FORMat {STANdard|MSB|LSB|TDM}

:BUS<n>:AUDio:FORMat?

- **Functional Description**

Set or query the data format for the AUDIO bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

{STANdard|MSB|LSB|TDM} represents standard, left-justified, right-justified, and time division multiplexing, respectively.

- **Return Format**

The query returns {STANdard|MSB|LSB|TDM}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:FORMat STANdard | Set the data format to STANdard. |
| :BUS1:AUDio:FORMat? | The query returns STANdard. |

### :BUS<n>:AUDio:ORDer

- **Command Format**

:BUS<n>:AUDio:ORDer {LSB|MSB}

:BUS<n>:AUDio:ORDer?

- **Functional Description**

Set or query the byte order for the AUDIO bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

LSB: Least significant bit

MSB: Most significant bit

- **Return Format**

The query returns {LSB|MSB}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:ORDer LSB | Set the byte order to LSB. |
| :BUS1:AUDio:ORDer? | The query returns LSB. |

### :BUS<n>:AUDio:BCLock:SOURce

- **Command Format**

:BUS<n>:AUDio:BCLock:SOURce <source>

:BUS<n>:AUDio:BCLock:SOURce?

- **Functional Description**

Set or query the source for the bit clock

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:BCLock:SOURce CHANnel1 | Set the source as Channel 1. |
| :BUS1:AUDio:BCLock:SOURce? | The query returns CHANnel1. |

### :BUS<n>:AUDio:WSELect:SOURce

- **Command Format**

  :BUS<n>:AUDio:WSELect:SOURce <source>

  :BUS<n>:AUDio:WSELect:SOURce?

- **Functional Description**

  Set or query the source for word selection.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:WSELect:SOURce CHANnel1 | Set the source as Channel 1. |
| :BUS1:AUDio:WSELect:SOURce? | The query returns CHANnel1. |

### :BUS<n>:AUDio:DATA:SOURce

- **Command Format**

  :BUS<n>:AUDio:DATA:SOURce <source>

  :BUS<n>:AUDio:DATA:SOURce?

- **Functional Description**

  Set or query the source for the data.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

  <Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:DATA:SOURce CHANnel1 | Set the source as Channel 1. |
| :BUS1:AUDio:DATA:SOURce? | The query returns CHANnel1. |

### :BUS<n>:AUDio:FSYNc:SOURce

- **Command Format**

:BUS<n>:AUDio:FSYNc:SOURce <source>

:BUS<n>:AUDio:FSYNc:SOURce?

- **Functional Description**

Set or query the source of the frame sync in TDM format.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:FSYNc:SOURce CHANnel1 | Set the source as Channel 1. |
| :BUS1:AUDio:FSYNc:SOURce? | The query returns CHANnel1. |

### :BUS<n>:AUDio:BCLock:LEVel

- **Command Format**

:BUS<n>:AUDio:BCLock:LEVel <level>

:BUS<n>:AUDio:BCLock:LEVel?

- **Functional Description**

Set or query the threshold for the bit clock.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Threshold value

- **Return Format**

The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:BCLock:LEVel 2 | Set the threshold to 2 V. |
| :BUS1:AUDio:BCLock:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:AUDio:WSELect:LEVel

- **Command Format**

  :BUS<n>:AUDio:WSELect:LEVel <level>

  :BUS<n>:AUDio:WSELect:LEVel?

- **Functional Description**

  Set or query the threshold for word selection.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:WSELect:LEVel 2 | Set the threshold to 2 V. |
| :BUS1:AUDio:WSELect:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:AUDio:DATA:LEVel

- **Command Format**

  :BUS<n>:AUDio:DATA:LEVel <level>

  :BUS<n>:AUDio:DATA:LEVel?

- **Functional Description**

  Set or query the threshold for bit data.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:DATA:LEVel 2 | Set the threshold to 2 V. |
| :BUS1:AUDio:DATA:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:AUDio:FSYNc:LEVel

- **Command Format**

  :BUS<n>:AUDio:FSYNc:LEVel <level>

:BUS<n>:AUDio:FSYNc:LEVel?

- **Functional Description**

Set or query the threshold of the frame sync in TDM format.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<level>: Threshold value

- **Return Format**

The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:FSYNc:LEVel 2 | Set the threshold to 2 V. |
| :BUS1:AUDio:FSYNc:LEVel? | The query returns 2.000000e+00. |

### :BUS<n>:AUDio:BCLock:POLarity

- **Command Format**

:BUS<n>:AUDio:BCLock:POLarity {POSitive|NEGative}

:BUS<n>:AUDio:BCLock:POLarity?

- **Functional Description**

Set or query the edge type of the bit clock for the AUDIO bus to POSitive (Rising) or NEGative (Falling).

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns the edge type { POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:BCLock:POLarity POS | Set the edge type of the bit clock to POSitive (Rising edge). |
| :BUS1:AUDio:BCLock:POLarity? | The query returns POSitive. |

### :BUS<n>:AUDio:WSELect:POLarity

- **Command Format**

:BUS<n>:AUDio:WSELect:POLarity {NORMal|INVert}

:BUS<n>:AUDio:WSELect:POLarity?

- **Functional Description**

Set or query the polarity of word selection for the AUDIO bus to NORMal (Normal) or INVert (Invert).

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns the edge type {NORMal|INVert}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:WSELect:POLarity NORMal | Set the polarity of word selection to NORMal. |
| :BUS1:AUDio:WSELect:POLarity? | The query returns NORMal. |

### :BUS<n>:AUDio:DATA:POLarity

- **Command Format**

:BUS<n>:AUDio:DATA:POLarity {H1|H0}

:BUS<n>:AUDio:DATA:POLarity?

- **Functional Description**

Set or query the data polarity for the AUDIO bus to H1 (H=1) or H0 (H=0).

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns the edge type {H1|H0}.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:DATA:POLarity H1 | Set the data polarity to H=1. |
| :BUS1:AUDio:DATA:POLarity? | The query returns H1. |

### :BUS<n>:AUDio:FSYNc:POLarity

- **Command Format**

:BUS<n>:AUDio:FSYNc:POLarity {POSitive|NEGative}

:BUS<n>:AUDio:FSYNc:POLarity?

- **Functional Description**

Set or query the polarity of the frame sync in TDM format to POSitive (Rising) or NEGative (Falling).

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

- **Return Format**

The query returns the edge type { POSitive | NEGative }.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:FSYNc:POLarity POS | Set the polarity of the frame sync to POSitive (Rising edge). |
| :BUS1:AUDio:FSYNc:POLarity? | The query returns POSitive. |

### :BUS<n>:AUDio:DLENgth

- **Command Format**

  :BUS<n>:AUDio:DLENgth <len>

  :BUS<n>:AUDio:DLENgth?

- **Functional Description**

  Set or query the bit size for the AUDIO bus.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <len>: Data length, ranging from 4 to 32.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

  :BUS1:AUDio:DLENgth 4 Set the data length of the data bits to 4.

  :BUS1:AUDio:DLENgth? The query returns 4.

### :BUS<n>:AUDio:TDM:CLOCk

- **Command Format**

  :BUS<n>:AUDio:TDM:CLOCk <val>

  :BUS<n>:AUDio:TDM:CLOCk?

- **Functional Description**

  Set or query the clock bit of each channel in TDM format for the AUDIO bus.

  <n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

  <val>: Clock bit, ranging from 4 to 32.

- **Return Format**

  The query returns the data length as an integer.

- **For Example**

  :BUS1:AUDio:TDM:CLOCk 4 Set the clock bit to 4.

  :BUS1:AUDio:TDM:CLOCk? The query returns 4.

### :BUS<n>:AUDio:TDM:NCHannel

- **Command Format**

  :BUS<n>:AUDio:TDM:NCHannel <val>

  :BUS<n>:AUDio:TDM:NCHannel?

- **Functional Description**

Set or query the channel number of each frame in TDM format for the AUDIO bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<val>: Channel number, ranging from 2 to 64.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:TDM:NCHannel 4 | Set the channel number to 4. |
| :BUS1:AUDio:TDM:NCHannel? | The query returns 4. |

### :BUS<n>:AUDio:TDM:DLENgth

- **Command Format**

:BUS<n>:AUDio:TDM:DLENgth <len>

:BUS<n>:AUDio:TDM:DLENgth?

- **Functional Description**

Set or query the data length of each channel in TDM format for the AUDIO bus.

<n>: {1|2|3|4} represents decoding 1, decoding 2, decoding 3, and decoding 4, respectively.

<len>: Data length, ranging from 4 to 32.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:TDM:DLENgth 4 | Set the data length of the data bits to 4. |
| :BUS1:AUDio:TDM:DLENgth? | The query returns 4. |

### :BUS<n>:AUDio:TDM:DELay

- **Command Format**

:BUS<n>:AUDio:TDM:DELay <val>

:BUS<n>:AUDio:TDM:DELay?

- **Functional Description**

Set or query the bit delay in TDM format for the AUDIO bus.

<val>: Bit delay, ranging from 0 to 31.

- **Return Format**

The query returns the data length as an integer.

- **For Example**

| | |
|---|---|
| :BUS1:AUDio:TDM:DELay 4 | Set the bit delay to 4. |

| | |
|---|---|
| :BUS1:AUDio:TDM:DELay? | The query returns 4. |

# AWG Command

This command is used Set the built-in signal source function on the oscilloscope.

## Channel Command

Set the channel function of the built-in signal source.

### AWG<n>:OUTPut

- **Command Format**

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

  AWG<n>:OUTPut?

- **Functional Description**

  Switch the output of the specified channel to ON or OFF.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the output stated of the specified channel the specified channel: 0 indicates OFF, while 1 indicates ON.

- **For Example**

| | |
|---|---|
| AWG1:OUTPut ON | Enable the output of AWG1. |
| AWG1:OUTPut? | The query returns 1. |

### AWG<n>:REVerse

- **Command Format**

  AWG<n>:REVerse {{1 | ON} | {0 | OFF}}

  AWG<n>:REVerse?

- **Functional Description**

  Switch the invert output of the specified channel to ON or OFF.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the invert output state of the specified channel: 0 indicates OFF, while 1 indicates ON.

- **For Example**

| | |
|---|---|
| AWG1:REVerse ON | Enable the invert output of AWG1. |

| | |
|---|---|
| AWG1:REVerse? | The query returns 1. |

### AWG<n>:LOAD

- **Command Format**

AWG<n>:LOAD <resistance>

AWG<n>:LOAD?

- **Functional Description**

Set the output load for the specified channel.

<resistance> represents the load resistance value, with the unit in Ω.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

Note: The load resistance ranges from 1-1,000,000, where 1,000,000 represents high resistance.

- **Return Format**

The query returns the load resistance value of the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:LOAD 50 | Set the output load for the AWG1 to 50 Ω. |
| AWG1:LOAD? | The query returns 5.000000e+01. |

### AWG<n>:MODe

- **Command Format**

AWG<n>:MODe { BASe | AM | FM | ASK | FSK | SWEep }

AWG<n>:MODe?

- **Functional Description**

Set the signal type for the specified channel to continuous wave, amplitude modulation, frequency modulation, amplitude shift keying, frequency shift keying, or sweep.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the signal type of the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:MODe BASe | Set the signal type for the AWG1 to BASe. |
| AWG1:MODe? | The query returns BASe. |

### AWG<n>:AMPLitude:UNIT

- **Command Format**

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

AWG<n>:AMPLitude:UNIT?

- **Functional Description**

Set or query the amplitude unit for the output of the specified channel.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the amplitude unit for the output of the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:AMPLitude:UNIT VPP | Set the amplitude unit for the AWG1 output to VPP. |
| AWG1:AMPLitude:UNIT? | The query returns VPP. |

### AWG<n>:BASe:WAVe

- **Command Format**

AWG<n>:BASe:WAVe { SINe | SQUare | PULSe | RAMP | ARB | NOISe | DC }

AWG<n>:BASe:WAVe?

- **Functional Description**

Set the fundamental type for the specified channel to sine wave, square wave, pulse wave, triangular wave, arbitrary wave, noise or DC.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the fundamental type of the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:BASe:WAVe SINe | Set the fundamental type of AWG1 to “SINe.” |
| AWG1:BASe:WAVe? | The query returns “SINe.” |

### AWG<n>:BASe:FREQuency

- **Command Format**

AWG<n>:BASe:FREQuency <freq>

AWG<n>:BASe:FREQuency?

- **Functional Description**

Set the output frequency for the specified channel.

<freq> represents the frequency value, with the unit in Hz, ranging from 1e to 6 Hz to the maximum frequency allowed by the current waveform.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the output frequency of the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:BASe:FREQuency 2000 | Set the output frequency of AWG1 to 2 kHz. |
| AWG1:BASe:FREQuency? | The query returns 2.000000e+003. |

### AWG<n>:BASe:PHASe

- **Command Format**

AWG<n>:BASe:PHASe <phase>

AWG<n>:BASe:PHAse?

- **Functional Description**

Set the output phase for the specified channel.

<phase> represents the phase value, with the unit in degrees (°), ranging from -360 to 360.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the output phase of the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:BASe:PHAse 20 | Set the output phase of AWG1 to 20°. |
| AWG1:BASe:PHAse? | The query returns 20. |

### AWG<n>:BASe:OFFSet

- **Command Format**

AWG<n>:BASe:OFFSet <voltage>

AWG<n>:BASe:OFFSet?

- **Functional Description**

Set the output DC offset for the specified channel.

<voltage> represents the voltage value, with the unit in V, ranging from 0 to the maximum DC under the current load.

The maximum DC under the current load = the current load*10/(50+ the current load) – the current minimum AC/2. The minimum AC is 2 mVpp; in DC mode, this value is considered as 0.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the DC offset of the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:BASe:OFFSet 2 | Set the output DC offset of AWG1 to 2 V. |

| | |
|---|---|
| AWG1:BASe:OFFSet? | The query returns 2.000000e+00. |

### AWG<n>:BASe:HIGH

- **Command Format**

  AWG<n>:BASe:HIGH <voltage>

  AWG<n>:BASe:HIGH?

- **Functional Description**

  Set the specified channel signal output high value, corresponding to the amplitude and DC offset.

  <voltage> represents the voltage value, with the specified unit of the current channel used as the unit.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the high voltage value of the specified channel signal in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:BASe:HIGH 2 | Set AWG1 to output the high value of 2 V. |
| AWG1:BASe:HIGH? | The query returns 2.000000e+00. |

### AWG<n>:BASe:LOW

- **Command Format**

  AWG<n>:BASe:LOW <voltage>

  AWG<n>:BASe:LOW?

- **Functional Description**

  Set the specified channel signal output low value, corresponding to the amplitude and DC offset.

  <voltage> represents the voltage value, with the specified unit of the current channel used as the unit.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the low voltage value of the specified channel signal in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:BASe:LOW 1 | Set AWG1 to output the low voltage value of 1 V. |
| AWG1:BASe:LOW? | The query returns 1.000000e+00. |

### AWG<n>:BASe:DUTY

- **Command Format**

  AWG<n>:BASe:DUTY <duty>

  AWG<n>:BASe:DUTY?

- **Functional Description**

  Set the output duty ratio for the specified channel.

  <duty> represents the duty ratio, with the unit in %, ranging from 0 to 100.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the output duty ratio of the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:BASe:DUTY 20 | Set the output duty ratio of AWG1 to 20%. |
| AWG1:BASe:DUTY? | The query returns 20. |

### AWG<n>:BASe:WIDTh

- **Command Format**

  AWG<n>:BASe:WIDTh <val>

  AWG<n>:BASe:WIDTh?

- **Functional Description**

  Set the output pulse width value for the specified channel's signal, corresponding to the duty ratio.

  <val> represents pulse width, with the unit in seconds (s).

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the pulse width value of the specified channel's signal.

### AWG<n>:RAMP:SYMMetry

- **Command Format**

  AWG<n>:RAMP:SYMMetry <symmetry>

  AWG<n>:RAMP:SYMMetry?

- **Functional Description**

  Set the output symmetry of slope signal for the specified channel.

  < symmetry > represents the symmetry, with the unit in %, ranging from 0 to 100.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the output symmetry of slope signal for the specified channel.

- **For Example**

  AWG1:RAMP:SYMMetry 20 Set the output symmetry of slope signal for the AWG1 to 20%.

  AWG1:RAMP:SYMMetry? The query returns 20.

### AWG<n>:PULSe:RISe

- **Command Format**

  AWG<n>:PULSe:RISe <width>

  AWG<n>:PULSe:RISe?

- **Functional Description**

  Set the pulse width of the rising edge for the specified channel's signal pulse waveform.

  <width> represents pulse width, with the unit in seconds (s).

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the pulse width of the rising edge for the specified channel's signal pulse waveform in scientific notation.

- **For Example**

  AWG1:PULSe:RISe 0.002 Set the pulse width of the rising edge for the AWG1 signal to 2 ms.

  AWG1:PULSe:RISe? The query returns 2e-3.

### AWG<n>:PULSe:FALL

- **Command Format**

  AWG<n>:PULSe:FALL <width>

  AWG<n>:PULSe:FALL?

- **Functional Description**

  Set the pulse width of the falling edge for the specified channel's signal pulse waveform.

  <width> represents pulse width, with the unit in seconds (s).

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the pulse width of the falling edge for the specified channel's signal pulse waveform in scientific notation.

- **For Example**

  AWG1:PULSe:FALL 0.002 Set the pulse width of the falling edge for the AWG1 signal to 2 ms.

| | |
|---|---|
| AWG1:PULSe:FALL? | The query returns 2e-3. |

### AWG<n>:MODulate:WAVe

- **Command Format**

  AWG<n>:MODulate:WAVe {SINe|SQUare|UPRamp|DNRamp|ARB|NOISe}

  AWG<n>:MODulate:WAVe?

- **Functional Description**

  Set the carrier signal type for the modulation signal of the specified channel to sine wave, square wave, upper triangular, lower triangular, arbitrary wave, or noise.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the carrier signal type for the modulation signal of the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:MODulate:WAVe SINe | Set the carrier signal type for the modulation signal of AWG1 to SINe. |
| AWG1:MODulate:WAVe? | The query returns SINe. |

### AWG<n>:MODulate:FREQuency

- **Command Format**

  AWG<n>:MODulate:FREQuency <freq>

  AWG<n>:MODulate:FREQuency?

- **Functional Description**

  Set the modulation frequency for the specified channel's signal.

  <freq> represents the frequency, with the unit in Hz.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the modulation frequency for the specified channel's signal in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:MODulate:FREQuency 2000 | Set the modulation frequency for the AWG1 signal to 2 kHz. |
| AWG1:MODulate:FREQuency? | The query returns 2.000000e+03. |

### AWG<n>:AM:MODulate:DEPTh

- **Command Format**

  AWG<n>:AM:MODulate:DEPTh <depth>

  AWG<n>:AM:MODulate:DEPTh?

- **Functional Description**

  Set the modulation depth of AM for the specified channel.

  <depth> represents the modulation depth, with the unit in %, ranging from 0% to 100%. The modulation depth of AM is 0% to 120%.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the modulation depth of AM for the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:AM:MODulate:DEPTh 50 | Set the modulation depth of AM for the AWG1 to 50%. |
| AWG1:AM:MODulate:DEPTh? | The query returns 50. |

### AWG<n>:FM:FREQuency:DEV

- **Command Format**

  AWG<n>:FM:FREQuency:DEV <freq>

  AWG<n>:FM:FREQuency:DEV?

- **Functional Description**

  Set the frequency offset in FM mode for the specified channel.

  <freq> represents the frequency offset, with the unit in Hz, ranging from 0 Hz to the current fundamental frequency.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the frequency offset of the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:FM:FREQuency:DEV 2000 | Set the frequency offset for the AWG1 to 2 kHz. |
| AWG1:FM:FREQuency:DEV? | The query returns 2.000000e+03. |

### AWG<n>:FSK:RATio

- **Command Format**

  AWG<n>:FSK:RATio <freq>

  AWG<n>:FSK:RATio?

- **Functional Description**

  Set the FSK rate for the specified channel.

  <freq> represents rate, with the unit in Hz.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the SK rate of the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:FSK:RATio 2000 | Set the FSK rate for the AWG1 to 2 kHz. |
| AWG1:FSK:RATio? | The query returns 2.000000e+03. |

### AWG<n>:FSK:FREQuency

- **Command Format**

  AWG<n>:FSK:FREQuency <freq>

  AWG<n>:FSK:FREQuency?

- **Functional Description**

  Set the hopping frequency in FSK mode for the specified channel.

  <freq> represents frequency, with the unit in Hz.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the hopping frequency of the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:FSK:FREQuency 2000 | Set the hopping frequency in FSK mode for the AWG1 to 2 kHz. |
| AWG1:FSK:FREQuency? | The query returns 2.000000e+003. |

### AWG<n>:SWEep:TYPe

- **Command Format**

  AWG<n>:SWEep:TYPe {LINe|LOG}

  AWG<n>:SWEep:TYPe?

- **Functional Description**

  Set or query the sweep mode for the specified channel to linear sweep or logarithmic sweep.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

  The query returns the sweep mode of the specified channel.

- **For Example**

| | |
|---|---|
| AWG1:SWEep:TYPe LINe | Set the sweep mode for the AWG1 to LINe. |
| AWG1:SWEep:TYPe? | The query returns LINe. |

### AWG<n>:SWEep:FREQuency:STARt

- **Command Format**

AWG<n>:SWEep:FREQuency:STARt <freq>

AWG<n>:SWEep:FREQuency:STARt?

- **Functional Description**

Set or query the start frequency in sweep mode for the specified channel.

<freq> represents the frequency value, with the unit in Hz, ranging from 1e to 6 Hz-the maximum allowed by the current waveform.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the start frequency in the sweep mode for the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:SWEep:FREQuency:STARt 2000 | Set the start frequency in the sweep mode of AWG1 to 2 kHz. |
| AWG1:SWEep:FREQuency:STARt? | The query returns 2.000000e+003. |

### AWG<n>:SWEep:FREQuency:STOP

- **Command Format**

AWG<n>:SWEep:FREQuency:STOP <freq>

AWG<n>:SWEep:FREQuency:STOP?

- **Functional Description**

Set or query the stop frequency in the sweep mode for the specified channel.

<freq> represents the frequency value, with the unit in Hz, ranging from 1e to 6 Hz to the maximum allowed by the current waveform.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the stop frequency in the sweep mode for the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:SWEep:FREQuency:STOP 2000 | Set the stop frequency in the sweep mode for the AWG1 to 2 kHz. |
| AWG1:SWEep:FREQuency:STOP? | The query returns 2.000000e+003. |

### AWG<n>:SWEep:TIMe

- **Command Format**

AWG<n>:SWEep:TIMe <time>

AWG<n>:SWEep:TIMe?

- **Functional Description**

Set or query the sweep time in the sweep mode for the specified channel.

<time> represents time, with the unit in seconds (s), ranging from 1 ms to 500s.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **Return Format**

The query returns the sweep time in the sweep mode for the specified channel in scientific notation.

- **For Example**

| | |
|---|---|
| AWG1:SWEep:TIMe 2 | Set the sweep time for the AWG1 to 2s. |
| AWG1:SWEep:TIMe? | The query returns 2.000000e+000. |

### AWG<n>:ARB

- **Command Format**

AWG<n>:ARB <filepath>

AWG<n>:ARB?

- **Functional Description**

Set the file path of the arbitrary wave for the specified channel.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

<filepath> represents the absolute file path of the file, which must be enclosed in double quotation marks as string data.

- **Return Format**

The query returns the file path of the arbitrary wave for the specified channel.

- **For Example**

AWG1:ARB "arbdir:/Common/AbsSine.bsv"

Set AWG1 Load the arbitray wave from the saved “AbsSine.bsv” file.

AWG1:ARB "udisk:/Common/AbsSine.bsv"

Set AWG1 Load the arbitray wave from the saved “AbsSine.bsv” file in a USB flash drive.

AWG1:ARB?

The query returns “arbdir:/Common/AbsSine.bsv."

### AWG<n>:MODulate:ARB

- **Command Format**

  AWG<n>:MODulate:ARB <filepath>

  AWG<n>:MODulate:ARB?

- **Functional Description**

  Set the file path of the modulation arbitrary wave for the specified channel.

  <n>: AWG serial number, where n can take the value 1, representing AWG1.

  <filepath> represents the absolute file path of the file, which must be enclosed in double quotation marks as string data.

- **Return Format**

  The query returns the file path of the modulation arbitrary wave for the specified channel.

- **For Example**

  AWG1:MODulate:ARB "arbdir:/Common/AbsSine.bsv"

  Set AWG1 Load the modulation arbitray wave from the saved “AbsSine.bsv” file.

  AWG1:MODulate:ARB "udisk:/Common/AbsSine.bsv"

  Set AWG1 Load the arbitray wave from the saved “AbsSine.bsv” file in a USB flash drive.

  AWG1:MODulate:ARB?

  The query returns “arbdir:/Common/AbsSine.bsv."

## Arbitrary Waveform Command

This command is used to write the arbitrary waveform file command, including write configuration of the fundamental arbitrary wave and modulation arbitrary wave.

### AWG<n>:WARB:MODulate

- **Command Format**

  AWG<n>:WARB:MODulate <arb file>

- **Functional Description**

  Write the modulation arbitrary wave, the maximum waveform data is 8k points. First, send this command first, and then send the arbitrary waveform file data to AWG.

  <arb file> represents the arbitrary wave file. This command only supports the data in .bsv

format.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **For Example**

AWG1:WARB:MODulate "test.bsv" Write the modulation arbitrary wave file for AWG1.

### AWG<n>:WARB:CARRier

- **Command Format**

AWG<n>:WARB:CARRier <arb file>

- **Functional Description**

Write the fundamental arbitrary wave, the maximum waveform data is 8k points. First, send this command first, and then send the arbitrary waveform file data to AWG.

<arb file> represents the arbitrary wave file. This command only supports the data in .bsv format.

<n>: AWG serial number, where n can take the value 1, representing AWG1.

- **For Example**

AWG1:WARB:CARRier "test.bsv" Write the fundamental arbitrary wave file for AWG1.

## LA Command

This command is used Set the digital channel function.

### :LA:STATe

- **Command Format**

:LA:STATe { {1|ON} | {0|OFF} }

:LA:STATe?

- **Functional Description**

Switch the LA function to ON or OFF, or query the state of the LA function.

- **Return Format**

The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

:LA:STATe ON Enable the LA function.

:LA:STATe? The query returns 1.

### :LA:ACTive

- **Command Format**

:LA:ACTive {<Dx> | OFF}

:LA:ACTive?

- **Functional Description**

Set or query the currently activated channel.

<Dx>: {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns the currently activated channel.

- **For Example**

| | |
|---|---|
| :LA:ACTive D3 | Set the currently activated channel to D3. |
| :LA:ACTive? | The query returns D3. |

### :LA:AUTOsort

- **Command Format**

:LA:AUTOsort { {1|ON} | {0|OFF} }

:LA:AUTOsort?

- **Functional Description**

Set the automatic sorting mode for the digital channel's waveform displayed on the screen.

OFF: The waveforms from top to bottom on the screen are D0 to D15.

ON: The waveforms from top to bottom on the screen are D15 to D0.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :LA:AUTOsort ON | The waveforms from top to bottom on the screen are D15 to D0. |
| :LA:AUTOsort? | The query returns 1. |

### :LA:DIGital<n>:DISPlay

- **Command Format**

:LA:DIGital<n>:DISPlay { {1|ON} | {0|OFF} }

:LA:DIGital<n>:DISPlay?

- **Functional Description**

Switch the specified single digital channel to ON or OFF, or query the state of the specified single digital channel.

<n>: An integer number from 1-16, representing 16 digital channels

{D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :LA:DIGital4:DISPlay ON | Enable D3. |
| :LA:DIGital4:DISPlay? | The query returns 1. |

### :LA:DIGital<n>:LABel

- **Command Format**

  :LA:DIGital<n>:LABel <label>

  :LA:DIGital<n>:LABel?

- **Functional Description**

  Set or query the lable of the specified single digital channel.

  <n>: An integer number from 1-16, representing

  {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

  <label>: ASCII string includes English letters, numbers, and some punctuation marks.

- **Return Format**

  The query returns the label of the specified single digital channel as an ASCII string.

- **For Example**

| | |
|---|---|
| :LA:DIGital3:LABel "ACK" | Set the label of D4 to ACK. |
| :LA:DIGital3:LABel? | The query returns ACK. |

### :LA:POD<n>:DISPlay

- **Command Format**

  :LA:POD<n>:DISPlay { {1|ON} | {0|OFF} }

  :LA:POD<n>:DISPlay?

- **Functional Description**

  Switch the specified default channel groups to ON or OFF, or query the state of the specified default channel groups.

  <n>: Integer number 1-2, POD1 represents (D0-D7), POD2 represents (D8-D15).

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :LA:POD1:DISPlay ON | Enable POD1 (D0-D7). |
| :LA:POD1:DISPlay? | The query returns 1. |

### :LA:POD<n>:THReshold

- **Command Format**

  :LA:POD<n>:THReshold <threshold>

  :LA:POD<n>:THReshold?

- **Functional Description**

  Set or query the threshold for the specified default channel groups. The default unit is V.

  <n>: Integer number 1-2, POD1 represents (D0-D7), POD2 represents (D8-D15).

  <threshold>: Real type, ranging from -20.0 V to +20.0 V.

- **Return Format**

  The query returns the threshold for the specified default channel groups in scientific notation.

- **For Example**

  :LA:POD1:THReshold 1.4 Set the threshold for POD1 (D0-D7) to 1.4 V.

  :LA:POD1:THReshold? The query returns 1.400000e+00.

### :LA:CHANnel<n>:THReshold

- **Command Format**

  :LA:CHANnel<n>:THReshold <threshold>

  :LA:CHANnel<n>:THReshold?

- **Functional Description**

  Set query the threshold for the specified default channel group. The default unit is V.

  <n>: Integer number 1, 2, CHANnel1 represents Channel 1, CHANnel2 represents Channel 2.

  <threshold>: Real type, the range is determined by the vertical scale.

- **Return Format**

  The query returns the threshold of the specified channel in scientific notation.

- **For Example**

  :LA:CHANnel1:THReshold 1.4 Set the threshold for Channel 1 to 1.4 V.

  :LA:CHANnel1:THReshold? The query returns 1.400000e+00.

### :LA:SIZe

- **Command Format**

  :LA:SIZe {SMALl|LARGe|MEDium}

  :LA:SIZe?

- **Functional Description**

Set or query the waveform size of the opened channel displayed on the screen.

- **Return Format**

The query returns SMALl, LARGe, or MEDium.

- **For Example**

| | |
|---|---|
| :LA:SIZE SMALl | Set the waveform size to SMALl. |
| :LA:SIZE? | The query returns SMALl. |

### :LA:TCALibrate

- **Command Format**

:LA:TCALibrate <tcal>

:LA:TCALibrate?

- **Functional Description**

Set or query the delay calibration time for the digital channel. The default unit is seconds (s).

<tcal>: Real type, ranging from -100 ns to 100 ns.

When using the oscilloscope to

When using an oscilloscope for actual measurements, the transmission delay of the probe cable can introduce significant errors (zero offset). Zero offset is defined as the offset of the waveform's intersection with the threshold voltage line relative to the trigger position. Users can correct the zero offset for the corresponding channel by setting a delay time.

- **Return Format**

The query returns the delay calibration time in scientific notation.

- **For Example**

| | |
|---|---|
| :LA:TCALibrate 20ns | Set the delay calibration time to 20 ns. |
| :LA:TCALibrate? | The query returns 2.000000E-8. |

### :LA:DELete

- **Command Format**

:LA:DELete {GROup1|GROup2|GROup3|GROup4}

:LA:DELete?

- **Functional Description**

Cancel the group setting of any one of the 16 digital channels, or to cancel the channel setting of any one of Group1-Group4. This command is only available for digital channels that have been grouped or to cancel the group setting of a customized channel group.

- **Return Format**

The query returns {GROup1|GROup2|GROup3|GROup4}.

- **For Example**

:LA:DELete GROup1 Cancel the channel setting of GROup1.

:LA:DELete? The query returns GROup1.

### :LA:GROup<n>:DISPlay

- **Command Format**

:LA:GROup<n>:DISPlay { {1|ON} | {0|OFF} }

:LA:GROup<n>:DISPlay?

- **Functional Description**

Switch the specified digital channel groups to ON or OFF.

<n>: An integer number from 1-4, representing GROup1, GROup2, GROup3, and GROup4, respectively.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

:LA:GROup1:DISPlay ON Enable the digital channel group 1.

:LA:GROup1:DISPlay? The query returns 1.

### :LA:GROup<n>:APPend

- **Command Format**

:LA:GROup<n>:APPend <digital0>[, digital1-digital15]

- **Functional Description**

Add a channel for the specified customized group.

<n>: An integer number from 1-4, representing GROup1, GROup2, GROup3, and GROup4, respectively.

<digital>: representing {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **For Example**

:LA:GROup1:APPend D0,D3 Add the channel D0 and D3 for group 1.

## LA Bus

This command is used Set LA bus data and serial decoding settings. The instrument can only function if equipped with this command.

### :LA:BUS<n>:DISPlay

- **Command Format**

  :LA:BUS<n>:DISPlay { {1|ON} | {0|OFF} }

  :LA:BUS<n>:DISPlay?

- **Functional Description**

  Set or query the LA function to ON or OFF on the oscilloscope.

  <n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :LA:BUS1:DISPlay ON | Enable LA bus 1 and display the decoded waveform. |
| :LA:BUS1:DISPlay? | The query returns 1. |

### :LA:BUS<n>:SOURce

- **Command Format**

  :LA:BUS<n>:SOURce <digital0>[, digital1- digital15, CHANnel1, CHANnel2]

  :LA:BUS<n>:SOURce?

- **Functional Description**

  Set the selected channel source of LA bus serial decoding on the oscilloscope.

  <n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

  <digital>: representing {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

  The query returns the currently selected channel source.

- **For Example**

| | |
|---|---|
| :LA:BUS1:SOURce D0,D3,CHANnel1 | Select the channel source D0, D3, and Channel 1 for the LA bus 1. |
| :LA:BUS1:SOURce? | The query returns D0, D3, CHANnel1. |

### :LA:BUS<n>:BASe

- **Command Format**

  :LA:BUS<n>:BASe {ASCII | BINary | HEX | DEC| WAV}

  :LA:BUS<n>:BASe?

- **Functional Description**

  Set the display format for the LA bus on the oscilloscope.

<n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

- **Return Format**

The query returns {ASCII | BINary | HEX | DEC | WAV }.

- **For Example**

| | |
|---|---|
| :LA:BUS1:BASe BIN | Set the display format for the LA bus 1 to BINary. |
| :LA:BUS1:BASe? | The query returns BINary. |

### :LA:BUS<n>:DATA?

- **Command Format**

:LA:BUS<n>:DATA?

- **Functional Description**

Read the data from the LA decoded event list on the oscilloscope.

- **Return Format**

The query returns the data from the LA decoded event list. The returned data conforms to the Data Block Format.

- **For Example**

:LA:BUS1:DATA? The query returns the data from the serial decoded event list for data bus 1:

#9000000089PAR,

ID,BUS1DATA,

0,0xFF

1,0xFE

2,0xF3

3,0xF5

4,0xF6

PAR represents the serial decoding, with the event table data in CSV format following. The specified format of the event data table is automatically adapted by different devices. The data are separated by commas and will automatically line wrap according to the list. The data value is related to the set decimal display format.

### :LA:BUS<n>:SLOPe

- **Command Format**

:LA:BUS<n>:SLOPe {POSitive|NEGative}

:LA:BUS<n>:SLOPe?

- **Functional Description**

Set the edge type of the clock channel when LA bus is sampling the data channel on the oscilloscope.

<n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

- **Return Format**

The query returns {POSitive|NEGative}.

- **For Example**

| | |
|---|---|
| :LA:BUS1:SLOPe POSitive | Set the clock source edge for the LA bus 1 to POSitive (Rising edge). |
| :LA:BUS1:SLOPe? | The query returns POSitive. |

### :LA:BUS<n>:CLK

- **Command Format**

:LA:BUS<n>:CLK {<Dx> | CHANnel1 | CHANnel2 | OFF}

:LA:BUS<n>:CLK?

- **Functional Description**

Set the clock source for the LA bus on the oscilloscope.

<n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

<Dx>: {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

- **Return Format**

The query returns {<Dx> | CHANnel1 | CHANnel2 | OFF}.

- **For Example**

| | |
|---|---|
| :LA:BUS1:CLK D0 | Set the clock source for the LA bus 1 to D0. |
| :LA:BUS1:CLK? | The query returns D0. |

### :LA:BUS<n>:POLarity

- **Command Format**

:LA:BUS<n>:POLarity {NEGative|POSitive}

:LA:BUS<n>:POLarity?

- **Functional Description**

Set the bit sequence, which determines the data polarity for the LA bus on the oscilloscope.

<n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

NEGative represents the bit sequence from high to low. POSitive represents the bit sequence from low to high.

- **Return Format**

The query returns {NEGative|POSitive}.

- **For Example**

| | |
|---|---|
| :LA:BUS1:POLarity POSitive | Set the bit sequence for the LA bus 1 to POSitive. |
| :LA:BUS1:POLarity? | The query returns POSitive. |

### :LA:BUS<n>:NREJect

- **Command Format**

:LA:BUS<n>:NREJect { {1|ON} | {0|OFF} }

:LA:BUS<n>:NREJect?

- **Functional Description**

Switch the jitter rejection function for the LA bus to ON or OFF on the oscilloscope.

<n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

- **Return Format**

The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :LA:BUS1:NREJect ON | Enable the jitter rejection function for the LA bus 1. |
| :LA:BUS1:NREJect? | The query returns 1. |

### :LA:BUS<n>:NRTime

- **Command Format**

:LA:BUS<n>:NRTime <time>

:LA:BUS<n>:NRTime?

- **Functional Description**

Set the jitter rejection time for the LA bus on the oscilloscope. The default unit is seconds (s).

<n>: Integer data {1|2} indicates the bus 1 or 2, respectively.

- **Return Format**

The query returns the noise rejection time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :LA:BUS1:NRTime 50us | Set the jitter rejection time for the LA bus to 50 us. |
| :LA:BUS1:NRTime? | The query returns 5.000000e-05. |

## BODe Command (Option)

This command is used Set the Bode plot function. The instrument can only function if equipped with this command.

### :BODe:APPLy

- **Command Format**

  :BODe:APPLy {{1 | ON} | {0 | OFF}}

  :BODe:APPLy?

- **Functional Description**

  Switch the Bode plot function to ON or OFF.

- **Return Format**

  The query returns Bode plot state: 0 indicates that Bode plot is disabled, while 1 indicates that the Bode plot is enabled.

- **For Example**

| | |
|---|---|
| :BODe:APPLy ON | Enable the Bode plot function. |
| :BODe:APPLy? | The query returns 1, indicating that the Bode plot function is enabled. |

### :BODe:MODe

- **Command Format**

  :BODe:MODe { BODe | PSRR }

  :BODe:MODe?

- **Functional Description**

  Set the Bode plot mode for the oscilloscope. BODe represents the Bode plot. PSRR represents power supply rejection ratio.

- **Return Format**

  The query returns { BODe | PSRR }.

- **For Example**

| | |
|---|---|
| :BODe:MODe PSRR | Set the Bode plot mode to PSRR. |
| :BODe:MODe? | The query returns PSRR. |

### :BODe:EVENt

- **Command Format**

  :BODe:EVENt {{1 | ON} | {0 | OFF}}

  :BODe:EVENt?

- **Functional Description**

  Enable or disable the event list for the Bode plot.

- **Return Format**

The query returns the event list status of Bode plot: 0 (disable) or 1 (enable).

- **For Example**

:BODe:EVENt ON Enable the event list for the Bode plot.

:BODe:EVENt? The query returns 1, indicating that the event list of the Bode plot is enabled.

### :BODe:EVENt:DATA?

- **Command Format**

:BODe:EVENt:DATA?

- **Functional Description**

Query the data from the event list of the Bode plot.

- **Return Format**

The query returns the data from the event list of the Bode plot. The list data is arranged in the CSV format, and the returned data conforms to the Data Block Format.

- **For Example**

:BODe:EVENt:DATA? The query returns the data from the event list of the Bode plot:

#9000000727BODE,

index,Freq,AMP(Vpp),Gain(dB),Phase(Â°) ,

1,3.700000e+03,3.000000e+00,-1.039986e+02,8.887821e+01,

2,8.784684e+03,3.000000e+00,-9.819112e+01,-7.619118e+01,

3,2.085694e+04,3.000000e+00,-9.908194e+01,-6.782645e+01

### :BODe:SWEep:MODe

- **Command Format**

:BODe:SWEep:MODe { SINGle | CONTinue}

:BODe:SWEep:MODe?

- **Functional Description**

Set the sweep mode for the Bode plot to SINGle (Single) or CONTinue (Continuous) on the oscilloscope.

- **Return Format**

The query returns { SINGle | CONTinue}.

- **For Example**

:BODe:SWEep:MODe SINGle Set the sweep mode for the Bode plot to SINGle (Single).

| | |
|---|---|
| :BODe:SWEep:MODe? | The query returns SINGle. |

### :BODe:SWEep:POINts

- **Command Format**

  :BODe:SWEep:POINts <points>

  :BODe:SWEep:POINts?

- **Functional Description**

  Set the sweep point for the Bode plot on the oscilloscope. <points> ranges from 1 to 1000.

- **Return Format**

  The query returns the sweep point.

- **For Example**

| | |
|---|---|
| :BODe:SWEep:POINts 1000 | Set the sweep point for the Bode plot to 1, 000. |
| :BODe:SWEep:POINts? | The query returns 1, 000. |

### :BODe:SWEep:FREQuency:STARt

- **Command Format**

  :BODe:SWEep:FREQuency:STARt <freq>

  :BODe:SWEep:FREQuency:STARt?

- **Functional Description**

  Set the start frequency for the Bode plot on the oscilloscope.

- **Return Format**

  The query returns the start frequency for the Bode plot on the oscilloscope, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :BODe:SWEep:FREQuency:STARt 1 kHz | Set the start frequency to 1 kHz. |
| :BODe:SWEep:FREQuency:STARt? | The query returns 1.000000e+03. |

### :BODe:SWEep:FREQuency:END

- **Command Format**

  :BODe:SWEep:FREQuency:END <freq>

  :BODe:SWEep:FREQuency:END?

- **Functional Description**

  Set the stop frequency for the Bode plot on the oscilloscope.

- **Return Format**

  The query returns the stop frequency for the Bode plot on the oscilloscope, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :BODe:SWEep:FREQuency:END 1 kHz | Set the stop frequency to 1 kHz. |
| :BODe:SWEep:FREQuency:END? | The query returns 1.000000e+03. |

### :BODe:AMPLitude:MODe

- **Command Format**

:BODe:AMPLitude:MODe { FIXed | VARiable }

:BODe:AMPLitude:MODe?

- **Functional Description**

Set the amplitude mode for the Bode plot FIXed (Fixed) or VARiable (Variable) on the oscilloscope.

- **Return Format**

The query returns { FIXed | VARiable }.

- **For Example**

| | |
|---|---|
| :BODe:AMPLitude:MODe FIXed | Set the amplitude mode for the Bode plot to FIXed. |
| :BODe:AMPLitude:MODe? | The query returns FIXed. |

### :BODe:AMPLitude<n>

- **Command Format**

:BODe:AMPLitude<n> <amp>

:BODe:AMPLitude<n>?

- **Functional Description**

Set the amplitude of the Bode plot on the oscilloscope. When the amplitude mode is FIXed, n can take the value of 1.

<n>: Ranging from 1 to 8

<amp>: Ranging from 20 mV to 6 V

- **Return Format**

The query returns 3.000000e-02, with the unit in V.

- **For Example**

| | |
|---|---|
| :BODe:AMPLitude1 30mV | Set the amplitude 1 of the Bode plot to 30 mV. |
| :BODe:AMPLitude1? | The query returns 3.000000e-02. |

### :BODe:SOURce:OFFSet

- **Command Format**

:BODe:SOURce:OFFSet <offset>

:BODe:SOURce:OFFSet?

- **Functional Description**

Set the DC offset of the source for the Bode plot on the oscilloscope.

<offset>: Ranging from -1 V to 1 V.

- **Return Format**

The query returns the DC offset of the source for the Bode plot on the oscilloscope, with the unit in V.

- **For Example**

| | |
|---|---|
| :BODe:SOURce:OFFSet 10mV | Set the DC offset of the source for the Bode plot to 10 mV. |
| :BODe:SOURce:OFFSet? | The query returns 1.000000e-02. |

### :BODe:SOURce:LOAD

- **Command Format**

:BODe:SOURce:LOAD {50OHM | HIGH}

:BODe:SOURce:LOAD?

- **Functional Description**

Set the resistance of the source for the Bode plot on the oscilloscope.

50OHM represents that the resistance is 50 Ω. HIGH represents high resistance.

- **Return Format**

The query returns {50OHM | HIGH}.

- **For Example**

| | |
|---|---|
| :BODe:SOURce:LOAD HIGH | Set the resistance of the source for the Bode plot to HIGH. |
| :BODe:SOURce:LOAD? | The query returns HIGH. |

### :BODe:DUT:INPut

- **Command Format**

:BODe:DUT:INPut {CHANnel1| CHANnel2| CHANnel3| CHANnel4}

:BODe:DUT:INPut?

- **Functional Description**

Set the DUT input channel for the Bode plot on the oscilloscope.

- **Return Format**

The query returns {CHANnel1| CHANnel2| CHANnel3| CHANnel4}.

- **For Example**

| | |
|---|---|
| :BODe:DUT:INPut CHANnel1 | Set the DUT input channel for the Bode plot to Channel 1. |
| :BODe:DUT:INPut? | The query returns CHANnel1. |

### :BODe:DUT:OUTPut

- **Command Format**

  :BODe:DUT:OUTPut {CHANnel1| CHANnel2| CHANnel3| CHANnel4}

  :BODe:DUT:OUTPut?

- **Functional Description**

  Set the DUT output channel for the Bode plot on the oscilloscope.

- **Return Format**

  The query returns {CHANnel1| CHANnel2| CHANnel3| CHANnel4}.

- **For Example**

| | |
|---|---|
| :BODe:DUT:OUTPut CHANnel1 | Set the DUT output channel for the Bode plot to Channel 1. |
| :BODe:DUT:OUTPut? | The query returns CHANnel1. |

### :BODe:AUTo

- **Command Format**

  :BODe:AUTo

- **Functional Description**

  Automatically set the Bode plot display.

- **For Example**

  :BODe:AUTo Automatically set the Bode plot display.

### :BODe:GAIN

- **Command Format**

  :BODe:GAIN {{1 | ON} | {0 | OFF}}

  :BODe:GAIN?

- **Functional Description**

  Enable or disable the gain of Bode plot.

- **Return Format**

  The query returns the gain status of Bode plot: 0 (disable) or 1 (enable).

- **For Example**

| | |
|---|---|
| :BODe:GAIN ON | Enable the gain of Bode plot. |
| :BODe:GAIN? | The query returns 1, indicating that the gain display of Bode plot is enabled. |

### :BODe:PHASe

- **Command Format**

  :BODe:PHASe {{1 | ON} | {0 | OFF}}

  :BODe:PHASe?

- **Functional Description**

  Enable or disable the phase of Bode plot.

- **Return Format**

  The query returns the phase status of Bode plot: 0 (disable) or 1 (enable).

- **For Example**

| | |
|---|---|
| :BODe:PHASe ON | Enable the phase of Bode plot. |
| :BODe:PHASe? | The query returns 1, indicating that the phase display of Bode plot is enabled. |

### :BODe:GAIN:SCALe

- **Command Format**

  :BODe:GAIN:SCALe <scale>

  :BODe:GAIN:SCALe?

- **Functional Description**

  Set the gain scale for Bode plot.

- **Return Format**

  The query returns the gain scale of Bode plot in scientific notation, with the unit in dB.

- **For Example**

| | |
|---|---|
| :BODe:GAIN:SCALe 2 | Set the gain scale for Bode plot to 2 dB. |
| :BODe:GAIN:SCALe? | The query returns 2.000000e+00. |

### :BODe:PHASe:SCALe

- **Command Format**

  :BODe:PHASe:SCALe <scale>

  :BODe:PHASe:SCALe?

- **Functional Description**

Set the phase scale for Bode plot.

- **Return Format**

The query returns the gain scale of Bode plot in scientific notation, with the unit in degrees (°).

- **For Example**

| | |
|---|---|
| :BODe:GAIN:SCALe 2 | Set the gain scale for Bode plot to 2°. |
| :BODe:GAIN:SCALe? | The query returns 2.000000e+00. |

### :BODe:PHASe:OFFSet

- **Command Format**

:BODe:PHASe:OFFSet <offset>

:BODe:PHASe:OFFSet?

- **Functional Description**

Set the phase offset for Bode plot.

- **Return Format**

The query returns the phase offset of the Bode plot in scientific notation, with the unit in degrees (°).

- **For Example**

| | |
|---|---|
| :BODe:PHASe:OFFSet -10 | Set the phase offset for Bode plot to -10°. |
| :BODe:PHASe:OFFSet? | The query returns -1.000000e+01. |

### :BODe:FREQuency:STARt

- **Command Format**

:BODe:FREQuency:STARt <freq>

:BODe:FREQuency:STARt?

- **Functional Description**

Set the start frequency for Bode plot.

- **Return Format**

The query returns the start frequency of Bode plot in scientific notation, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :BODe:FREQuency:STARt 1kHz | Set the start frequency for Bode plot to 1 kHz. |
| :BODe:FREQuency:STARt? | The query returns 1.000000e+03. |

### :BODe:FREQuency:END

- **Command Format**

:BODe:FREQuency:END <freq>

:BODe:FREQuency:END?

- **Functional Description**

  Set the stop frequency for Bode plot.

- **Return Format**

  The query returns the stop frequency of Bode plot in scientific notation, with the unit in Hz.

- **For Example**

| | |
|---|---|
| :BODe:FREQuency:END 2kHz | Set the stop frequency for Bode plot to 2 kHz. |
| :BODe:FREQuency:END? | The query returns 2.000000e+03. |

### :BODe:PHASe:MARGin?

- **Command Format**

  :BODe:PHASe:MARGin?

- **Functional Description**

  Set the phase margin in a Bode plot represents the phase change that can be added before the system becomes unstable. A larger phase margin indicates a more stable system, but it may also result in slower system response.

- **Return Format**

  The query returns the phase margin of the Bode plot.

- **For Example**

| | |
|---|---|
| :BODe:PHASe:MARGin? | The query returns 110.3°@50 Hz. |

### :BODe:GAIN:MARGin?

- **Command Format**

  :BODe:GAIN:MARGin?

- **Functional Description**

  Set the gain margin in a Bode plot. A positive gain margin indicates that the system is stable, while a negative gain margin indicates that the system is unstable.

- **Return Format**

  The query returns the gain margin in a Bode plot.

- **For Example**

| | |
|---|---|
| :BODe:GAIN:MARGin? | The query returns 52.11dB@114 Hz. |

## SEARch Command

This command is used to control the search function on the oscilloscope.

### :SEARch:ENABle

- **Command Format**

  :SEARch:ENABle { {1|ON} | {0|OFF} }

  :SEARch:ENABle?

- **Functional Description**

  Set or query the search function to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :SEARch:ENABle ON | Enable the search function. |
| :SEARch:ENABle? | The query returns 1. |

### :SEARch:COPY

- **Command Format**

  :SEARch:COPY {STOT|TTOS}

- **Functional Description**

  Set the copy function under the search.

  STOT: Copy to trigger

  TTOS: Copy from trigger.

- **For Example**

| | |
|---|---|
| :SEARch:COPY STOT | Execute STOT. |

### :SEARch:MODe

- **Command Format**

  :SEARch:MODe <mode>

  :SEARch:MODe?

- **Functional Description**

  Set or query the search type.

  <mode>:

  {EDGe|PULSe|SLOPe|RUNT|WINDow|DELay|TIMeout|DURation|SHOLd|NEDGe|PATTern}

  represents EDGe (Edge), PULSe (Pulse width), SLOPe (Slope), RUNT (Runt), WINDow (Over-amplitude), DELay (Delay), TIMeout (Tiemout), DURation (Duration), SHOLd (Setup & Hold), NEDGe (Nth edge), and PATTern (Code pattern), respectively.

- **Return Format**

The query returns the search type.

- **For Example**

| | |
|---|---|
| :SEARch:MODe NEDGe | Set the search type to NEDGe. |
| :SEARch:MODe? | The query returns NEDGe. |

### :SEARch:EVENt

- **Command Format**

:SEARch:EVENt { {1|ON} | {0|OFF} }

:SEARch:EVENt?

- **Functional Description**

Set or query the event list of the search to ON or OFF.

- **Return Format**

The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :SEARch:EVENt ON | Enable the event list of the search. |
| :SEARch:EVENt? | The query returns 1. |

### :SEARch:EVENt:DATA?

- **Command Format**

:SEARch:EVENt:DATA?

- **Functional Description**

Query the data from the event list of the search.

- **Return Format**

The query returns the data from the event list of the search. The list data is arranged in the CSV format, and the returned data conforms to the Data Block Format.

- **For Example**

:SEARch:EVENt:DATA? The query returns the data from the event list of the search:

#900000072SEARH,

index,time,

1,3.700000e-03,

2,8.784684e-03,

3,2.085694e-04

## Edge Search

### :SEARch:EDGe:SOURce

- **Command Format**

  :SEARch:EDGe:SOURce <source>

  :SEARch:EDGe:SOURce?

- **Functional Description**

  Set or query the search source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n>: Physical channel

- **Return Format**

  The query returns the search source {CHANnel1|CHANnel2|CHANnel3|CHANnel4|}.

- **For Example**

| | |
|---|---|
| :SEARch:EDGe:SOURce CHANnel1 | Set the search source as Channel 1. |
| :SEARch:EDGe:SOURce? | The query returns CHANnel1. |

### :SEARch:EDGe:LEVel

- **Command Format**

  :SEARch:EDGe:LEVel <level>

  :SEARch:EDGe:LEVel?

- **Functional Description**

  Set or query the threshold.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:EDGe:LEVel 2 | Set the threshold to 2 V. |
| :SEARch:EDGe:LEVel? | The query returns 2.000000e+00. |

### :SEARch:EDGe:POLarity

- **Command Format**

  :SEARch:EDGe:POLarity {POSitive|NEGative|ANY}

  :SEARch:EDGe:POLarity?

- **Functional Description**

Set the edge type for the edge search to POSitive (Rising edge), NEGative (Falling edge), or ANY (Arbitrary edge).

- **Return Format**

The query returns the edge type of the search { POSitive | NEGative | ANY }.

- **For Example**

| | |
|---|---|
| :SEARch:EDGe:POLarity POS | Set the edge type for the edge search to POSitive (Rising edge). |
| :SEARch:EDGe:POLarity? | The query returns POSitive. |

## Pulse Width Search

### :SEARch:PULSe:SOURce

- **Command Format**

:SEARch:PULse:SOURce <source>

:SEARch:PULse:SOURce?

- **Functional Description**

Set or query the search source.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

CHANnel<n>: Physical channel

- **Return Format**

The query returns the search source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:PULse:SOURce CHANnel1 | Set the search source as Channel 1. |
| :SEARch:PULse:SOURce? | The query returns CHANnel1. |

### :SEARch:PULSe:LEVel

- **Command Format**

:SEARch:PULse:LEVel <level>

:SEARch:PULse:LEVel?

- **Functional Description**

Set or query the threshold.

<level>: Threshold value

- **Return Format**

The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:PULse:LEVel 2 | Set the threshold to 2 V. |
| :SEARch:PULse:LEVel? | The query returns 2.000000e+00. |

### :SEARch:PULSe:QUALifier

- **Command Format**

:SEARch:PULSe:QUALifier {GREaterthan | LESSthan | INRange}

:SEARch:PULSe:QUALifier?

- **Functional Description**

Set the time search condition for the pulse search to GREaterthan (Greater than), LESSthan (Less than), or INRange (Within the range).

- **Return Format**

The query returns {GREaterthan | LESSthan | INRange}.

- **For Example**

| | |
|---|---|
| :SEARch:PULSe:QUALifier GRE | Set the pulse condition to GREaterthan. |
| :SEARch:PULSe:QUALifier? | The query returns GREaterthan. |

### :SEARch:PULSe:POLarity

- **Command Format**

:SEARch:PULSe:POLarity {POSitive | NEGative}

:SEARch:PULSe:POLarity?

- **Functional Description**

Set the pulse polarity to POSitive (Positive pulse width) or NEGative (Negative pulse width).

- **Return Format**

The query returns { POSitive | NEGative }.

- **For Example**

| | |
|---|---|
| :SEARch:PULSe:POL POS | Set the pulse polarity to POSitive (Positive pulse width). |
| :SEARch:PULSe:POL? | The query returns POSitive. |

### :SEARch:PULSe:TIMe:UPPer

- **Command Format**

:SEARch:PULSe:TIMe:UPPer <time>

:SEARch:PULSe:TIMe:UPPer?

- **Functional Description**

Set the upper limit time for the pulse width search.

- **Return Format**

The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:PULSe:TIMe:UPPer 1 | Set the upper limit time for the pulse width search to 1s. |
| :SEARch:PULSe:TIMe:UPPer? | The query returns 1.000000e+00. |

### :SEARch:PULSe:TIMe:LOWer

- **Command Format**

:SEARch:PULSe:TIMe:LOWer <time>

:SEARch:PULSe:TIMe:LOWer?

- **Functional Description**

Set the lower limit time for the pulse width search.

- **Return Format**

The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:PULSe:TIMe:LOWer 1 | Set the lower limit time for the pulse width search to 1s. |
| :SEARch:PULSe:TIMe:LOWer? | The query returns 1.000000e+00. |

## Slope Search

### :SEARch:SLOPe:SOURce

- **Command Format**

:SEARch:SLOPe:SOURce <source>

:SEARch:SLOPe:SOURce?

- **Functional Description**

Set or query the search source.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

The query returns the search source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:SLOPe:SOURce CHANnel1 | Set the search source as Channel 1. |
| :SEARch:SLOPe:SOURce? | The query returns CHANnel1. |

### :SEARch:SLOPe:LOW:LEVel

- **Command Format**

  :SEARch:SLOPe:LOW:LEVel <level>

  :SEARch:SLOPe:LOW:LEVel?

- **Functional Description**

  Set or query the low threshold value.

  <level>: Low threshold value

- **Return Format**

  The query returns low threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:SLOPe:LOW:LEVel -3 | Set the low threshold value to -3 V. |
| :SEARch:SLOPe:LOW:LEVel? | The query returns -3.000000e+00. |

### :SEARch:SLOPe:HIGH:LEVel

- **Command Format**

  :SEARch:SLOPe:HIGH:LEVel <level>

  :SEARch:SLOPe:HIGH:LEVel?

- **Functional Description**

  Set or query the high threshold value.

  <level>: High threshold value

- **Return Format**

  The query returns the high threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:SLOPe:HIGH:LEVel 3 | Set the high threshold value to 3 V. |
| :SEARch:SLOPe:HIGH:LEVel? | The query returns 3.000000e+00. |

### :SEARch:SLOPe:QUALifier

- **Command Format**

  :SEARch:SLOPe:QUALifier {GREaterthan | LESSthan | INRange}

  :SEARch:SLOPe:QUALifier?

- **Functional Description**

  Set the time search condition for the slope to GREaterthan (Greater than), LESSthan (Less than),

or INRange (Within the range).

- **Return Format**

The query returns {GREaterthan | LESSthan | INRange}.

- **For Example**

:SEARch:SLOPe:QUALifier GRE Set the slope condition to GREaterthan.

:SEARch:SLOPe:QUALifier? The query returns GREaterthan.

### :SEARch:SLOPe:POLarity

- **Command Format**

:SEARch:SLOPe:POLarity {POSitive|NEGative}

:SEARch:SLOPe:POLarity?

- **Functional Description**

Set the edge type for the slope search to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns {POSitive|NEGative}.

- **For Example**

:SEARch:SLOPe:POLarity POS Set the edge type for the slope search to POSitive (Rising edge).

:SEARch:SLOPe:POLarity? The query returns POSitive.

### :SEARch:SLOPe:TIMe:UPPer

- **Command Format**

:SEARch:SLOPe:TIMe:UPPer <time>

:SEARch:SLOPe:TIMe:UPPer?

- **Functional Description**

Set the upper limit time for the slope search.

- **Return Format**

The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

:SEARch:SLOPe:TIMe:UPPer 1 Set the upper limit time for the slope search to 1s.

:SEARch:SLOPe:TIMe:UPPer? The query returns 1.000000e+00.

### :SEARch:SLOPe:TIMe:LOWer

- **Command Format**

:SEARch:SLOPe:TIMe:LOWer <time>

:SEARch:SLOPe:TIMe:LOWer?

- **Functional Description**

Set the lower limit time for the slope search.

- **Return Format**

The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:SLOPe:TIMe:LOWer 1 | Set the lower limit time for the slope search to 1s. |
| :SEARch:SLOPe:TIMe:LOWer? | The query returns 1.000000e+00. |

## Runt Search

### :SEARch:RUNT:SOURce

- **Command Format**

:SEARch:RUNT:SOURce <source>

:SEARch:RUNT:SOURce?

- **Functional Description**

Set or query the search source.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

The query returns the search source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:RUNT:SOURce CHANnel1 | Set the search source as Channel 1. |
| :SEARch:RUNT:SOURce? | The query returns CHANnel1. |

### :SEARch:RUNT:LOW:LEVel

- **Command Format**

:SEARch:RUNT:LOW:LEVel <level>

:SEARch:RUNT:LOW:LEVel?

- **Functional Description**

Set or query the low threshold value.

<level>: Low threshold value

- **Return Format**

The query returns low threshold in scientific notation. The unit conforms to the current

amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:RUNT:LOW:LEVel -3 | Set the low threshold value to -3 V. |
| :SEARch:RUNT:LOW:LEVel? | The query returns -3.000000e+00. |

### :SEARch:RUNT:HIGH:LEVel

- **Command Format**

:SEARch:RUNT:HIGH:LEVel <level>

:SEARch:RUNT:HIGH:LEVel?

- **Functional Description**

Set or query the high threshold value.

<level>: High threshold value

- **Return Format**

The query returns the high threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:RUNT:HIGH:LEVel 3 | Set the high threshold value to 3 V. |
| :SEARch:RUNT:HIGH:LEVel? | The query returns 3.000000e+00. |

### :SEARch:RUNT:QUALifier

- **Command Format**

:SEARch:RUNT:QUALifier {GREaterthan | LESSthan | INRange | NONE}

:SEARch:RUNT:QUALifier?

- **Functional Description**

Set the time condition for the runt search to GREaterthan (Greater than), LESSthan (Less than), INRange (Within the range), OUTRange (Outside of the range), or NONE (Arbitrary).

- **Return Format**

The query returns {GREaterthan | LESSthan | INRange | NONE}.

- **For Example**

| | |
|---|---|
| :SEARch:RUNT:QUALifier GRE | Set the runt condition to GREaterthan. |
| :SEARch:RUNT:QUALifier? | The query returns GREaterthan. |

### :SEARch:RUNT:POLarity

- **Command Format**

:SEARch:RUNT:POLarity {POSitive | NEGative}

:SEARch:RUNT:POLarity?

- **Functional Description**

Set the polarity for the runt search to POSitive (Positive pulse width) or NEGative (Negative pulse width).

- **Return Format**

The query returns {POSitive | NEGative}.

- **For Example**

| | |
|---|---|
| :SEARch:RUNT:POL POS | Set the pulse polarity to POSitive (Positive pulse width). |
| :SEARch:RUNT:POL? | The query returns POSitive. |

### :SEARch:RUNT:TIMe:UPPer

- **Command Format**

:SEARch:RUNT:TIMe:UPPer <time>

:SEARch:RUNT:TIMe:UPPer?

- **Functional Description**

Set the upper limit time for the runt search.

- **Return Format**

The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:RUNT:TIMe:UPPer 1 | Set the upper limit time for the runt search to 1s. |
| :SEARch:RUNT:TIMe:UPPer? | The query returns 1.000000e+00. |

### :SEARch:RUNT:TIMe:LOWer

- **Command Format**

:SEARch:RUNT:TIMe:LOWer <time>

:SEARch:RUNT:TIMe:LOWer?

- **Functional Description**

Set the lower limit time for the runt search.

- **Return Format**

The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:RUNT:TIMe:LOWer 1 | Set the lower limit time for the runt search to 1s. |
| :SEARch:RUNT:TIMe:LOWer? | The query returns 1.000000e+00. |

## Over-amplitude Search

### :SEARch:WINDow:SOURce

- **Command Format**

  :SEARch:WINDow:SOURce <source>

  :SEARch:WINDow:SOURce?

- **Functional Description**

  Set or query the search source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

  The query returns the search source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:WINDow:SOURce CHANnel1 | Set the search source as Channel 1. |
| :SEARch:WINDow:SOURce? | The query returns CHANnel1. |

### :SEARch:WINDow:LOW:LEVel

- **Command Format**

  :SEARch:WINDow:LOW:LEVel <level>

  :SEARch:WINDow:LOW:LEVel?

- **Functional Description**

  Set or query the low threshold value.

  <level>: Low threshold value

- **Return Format**

  The query returns low threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:WINDow:LOW:LEVel -3 | Set the low threshold value to -3 V. |
| :SEARch:WINDow:LOW:LEVel? | The query returns -3.000000e+00. |

### :SEARch:WINDow:HIGH:LEVel

- **Command Format**

  :SEARch:WINDow:HIGH:LEVel <level>

  :SEARch:WINDow:HIGH:LEVel?

- **Functional Description**

  Set or query the high threshold value.

  <level>: High threshold value

- **Return Format**

  The query returns the high threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:WINDow:HIGH:LEVel 3 | Set the high threshold value to 3 V. |
| :SEARch:WINDow:HIGH:LEVel? | The query returns 3.000000e+00. |

### :SEARch:WINDow:POLarity

- **Command Format**

  :SEARch:WINDow:POLarity {POSitive|NEGative|ANY}

  :SEARch:WINDow:POLarity?

- **Functional Description**

  Set the edge type for the window search to POSitive (Rising edge), NEGative (Falling edge), or ANY (Arbitrary).

- **Return Format**

  The query returns the edge type of the search {POSitive|NEGative|ANY}.

- **For Example**

| | |
|---|---|
| :SEARch:WINDow:POLarity POS | Set the edge type for the window search to POSitive (Rising edge). |
| :SEARch:WINDow:POLarity? | The query returns POSitive. |

### :SEARch:WINDow:TIMe

- **Command Format**

  :SEARch:WINDow:TIMe <time>

  :SEARch:WINDow:TIMe?

- **Functional Description**

  Set the time interval for the window search.

- **Return Format**

  The query returns the current time interval, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:WINDow:TIMe 1 | Set the time interval for the window search to 1s. |

| | |
|---|---|
| :SEARch:WINDow:TIMe? | The query returns 1.000000e+00. |

### :SEARch:WINDow:POSition

- **Command Format**

  :SEARch:WINDow:POSition {ENTer|EXIT|TIMe}

  :SEARch:WINDow:POSition?

- **Functional Description**

  Set the position for the window search.

- **Return Format**

  The query returns {ENTer|EXIT|TIMe}.

- **For Example**

| | |
|---|---|
| :SEARch:WINDow:POS TIMe | Set the position for the window search to TIMe. |
| :SEARch:WINDow:POS? | The query returns TIMe. |

## Delay Search

### :SEARch:DELay:ASOURce

- **Command Format**

  :SEARch:DELay:ASOURce {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}

  :SEARch:DELay:ASOURce?

- **Functional Description**

  Set the source 1 for the delay search.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:DELay:ASOURce CHAN1 | Set the search source 1 to Channel 1. |
| :SEARch:DELay:ASOURce? | The query returns CHANnel1. |

### :SEARch:DELay:ALEVel

- **Command Format**

  :SEARch:DELay:ALEVel <level>

  :SEARch:DELay:ALEVel?

- **Functional Description**

  Set or query the threshold for the source 1.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:DELay:ALEVel 2 | Set the threshold of the source 1 to 2 V. |
| :SEARch:DELay:ALEVel? | The query returns 2.000000e+00. |

### :SEARch:DELay:APOLarity

- **Command Format**

  :SEARch:DELay:APOLarity {NEGative | POSitive}

  :SEARch:DELay:APOLarity?

- **Functional Description**

  Set the edge type for the search source 1 to POSitive (Rising) or NEGative (Falling).

- **Return Format**

  The query returns {NEGative | POSitive}.

- **For Example**

| | |
|---|---|
| :SEARch:DELay:APOLarity NEG | Set the edge type for the search source 1 to NEGative. |
| :SEARch:DELay:APOLarity? | The query returns NEGative. |

### :SEARch:DELay:BSOURce

- **Command Format**

  :SEARch:DELay:BSOURce {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}

  :SEARch:DELay:BSOURce?

- **Functional Description**

  Set the source 2 for the delay search.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:DELay:BSOURce CHAN1 | Set the source 2 to Channel 1. |
| :SEARch:DELay:BSOURce? | The query returns CHANnel1. |

### :SEARch:DELay:BLEVel

- **Command Format**

  :SEARch:DELay:BLEVel <level>

:SEARch:DELay:BLEVel?

- **Functional Description**

Set or query the threshold for the source 2.

<level>: Threshold value

- **Return Format**

The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:DELay:BLEVel 2 | Set the threshold of the source 2 to 2 V. |
| :SEARch:DELay:BLEVel? | The query returns 2.000000e+00. |

### :SEARch:DELay:BPOLarity

- **Command Format**

:SEARch:DELay:BPOLarity {NEGative | POSitive}

:SEARch:DELay:BPOLarity?

- **Functional Description**

Set the edge type for the search source 2 to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns {NEGative | POSitive}.

- **For Example**

| | |
|---|---|
| :SEARch:DELay:BPOLarity NEG | Set the edge type for the search source 2 to NEGative. |
| :SEARch:DELay:BPOLarity? | The query returns NEGative. |

### :SEARch:DELay:QUALifier

- **Command Format**

:SEARch:DELay:QUALifier { GREaterthan | LESSthan | INRange | OUTRange }

:SEARch:DELay:QUALifier?

- **Functional Description**

Set the time interval condition for the delay search to GREaterthan (Greater than), LESSthan (Less than), INRange (Within the range), or NONE (Arbitrary).

- **Return Format**

The query returns { GREaterthan | LESSthan | INRange | OUTRange }.

- **For Example**

| | |
|---|---|
| :SEARch:DELay:QUALifier GRE | Set the delay condition to GREaterthan. |

| | |
|---|---|
| :SEARch:DELay:QUALifier? | The query returns GREaterthan. |

### :SEARch:DELay:TIMe:UPPer

- **Command Format**

  :SEARch:DELay:TIMe:UPPer <time>

  :SEARch:DELay:TIMe:UPPer?

- **Functional Description**

  Set the upper limit time for the delay search.

- **Return Format**

  The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:DELay:TIMe:UPPer 1 | Set the upper limit time for the delay search to 1s. |
| :SEARch:DELay:TIMe:UPPer? | The query returns 1.000000e+00. |

### :SEARch:DELay:TIMe:LOWer

- **Command Format**

  :SEARch:DELay:TIMe:LOWer <time>

  :SEARch:DELay:TIMe:LOWer?

- **Functional Description**

  Set the lower limit time for the delay search.

- **Return Format**

  The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:DELay:TIMe:LOWer 1 | Set the lower limit time for the delay search to 1s. |
| :SEARch:DELay:TIMe:LOWer? | The query returns 1.000000e+00. |

## Timeout Search

### :SEARch:TIMeout:SOURce

- **Command Format**

  :SEARch:TIMeout:SOURce <source>

  :SEARch:TIMeout:SOURce?

- **Functional Description**

  Set or query the search source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

The query returns the search source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:TIMeout:SOURce CHANnel1 | Set the search source as Channel 1. |
| :SEARch:TIMeout:SOURce? | The query returns CHANnel1. |

### :SEARch:TIMeout:LEVel

- **Command Format**

:SEARch:TIMeout:LEVel <level>

:SEARch:TIMeout:LEVel?

- **Functional Description**

Set or query the threshold.

<level>: Threshold value

- **Return Format**

The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:TIMeout:LEVel -3 | Set the threshold to -3 V. |
| :SEARch:TIMeout:LEVel? | The query returns -3.000000e+00. |

### :SEARch:TIMeout:TIMe

- **Command Format**

:SEARch:TIMeout:TIMe <time>

:SEARch:TIMeout:TIMe?

- **Functional Description**

Set the time interval for the timeout search.

- **Return Format**

The query returns the current time interval, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:TIMeout:TIMe 1 | Set the time interval for the timeout search to 1s. |
| :SEARch:TIMeout:TIMe? | The query returns 1.000000e+00. |

### :SEARch:TIMeout:POLarity

- **Command Format**

  :SEARch:TIMeout:POLarity {POSitive|NEGative|ANY}

  :SEARch:TIMeout:POLarity?

- **Functional Description**

  Set the edge type for the timeout search to POSitive (Rising edge), NEGative (Falling edge), or ANY (Arbitrary).

- **Return Format**

  The query returns the edge type of the timeout search {POSitive|NEGative|ANY}.

- **For Example**

| | |
|---|---|
| :SEARch:TIMeout:POLarity POS | Set the edge type to POSitive (Rising edge). |
| :SEARch:TIMeout:POLarity? | The query returns POSitive. |

## Duration Search

### :SEARch:DURation:LEVel

- **Command Format**

  :SEARch:DURation:LEVel <level>

  :SEARch:DURation:LEVel?

- **Functional Description**

  Set or query the threshold.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:DURation:LEVel 3 | Set the threshold to 3 V. |
| :SEARch:DURation:LEVel? | The query returns 3.000000e+00. |

### :SEARch:DURation:PATTern

- **Command Format**

  :SEARch:DURation:PATTern <source>,<pch>

  :SEARch:DURation:PATTern? <source>

- **Functional Description**

  Set or query the trigger code pattern for the specified source. X represents the default value.

<source>: {CHANnel1 | CHANnel2| CHANnel3| CHANnel4 | <Dx>}.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

<pch>: {H|L|X}.

- **Return Format**

The query returns the current trigger code pattern of the specified source.

- **For Example**

| | |
|---|---|
| :SEARch:DURation:PATTern CHANnel1,H | Set the code pattern of Channel 1 as H. |
| :SEARch:DURation:PATTern? CHANnel1 | The query returns H. |

### :SEARch:DURation:QUALifier

- **Command Format**

:SEARch:DURation:QUALifier { GREaterthan | LESSthan | INRange }

:SEARch:DURation:QUALifier?

- **Functional Description**

Set the time interval for the delay search to GREaterthan (Greater than), LESSthan (Less than), or INRange (Within the range).

- **Return Format**

The query returns { GREaterthan | LESSthan | INRange }.

- **For Example**

| | |
|---|---|
| :SEARch:DURation:QUALifier GRE | Set the slope condition to GREaterthan. |
| :SEARch:DURation:QUALifier? | The query returns GREaterthan. |

### :SEARch:DURation:TIMe:LOWer

- **Command Format**

:SEARch:DURation:TIMe:LOWer <time>

:SEARch:DURation:TIMe:LOWer?

- **Functional Description**

Set the lower limit time for the duration search. The lower limit time can be set when the time interval condition is “GREaterthan.”

- **Return Format**

The query returns the lower limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:DURation:TIMe:LOWer 1 | Set the lower limit time for the duration search to 1s. |
| :SEARch:DURation:TIMe:LOWer? | The query returns 1.000000e+00. |

### :SEARch:DURation:TIMe:UPPer

- **Command Format**

  :SEARch:DURation:TIMe:UPPer <time>

  :SEARch:DURation:TIMe:UPPer?

- **Functional Description**

  Set the upper limit time for the duration search. The upper limit time can be set when the time interval condition is LESSthan.

- **Return Format**

  The query returns the upper limit of the current time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:DURation:TIMe:UPPer 1 | Set the upper limit time for the duration search to 1s. |
| :SEARch:DURation:TIMe:UPPer? | The query returns 1.000000e+00. |

## Setup & Hold Search

### :SEARch:SHOLd:SDA

- **Command Format**

  :SEARch:SHOLd:SDA {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}

  :SEARch:SHOLd:SDA?

- **Functional Description**

  Set the data source for the setup & hold search.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:SHOLd:SDA CHAN1 | Set Channel 1 as data source. |
| :SEARch:SHOLd:SDA? | The query returns CHANnel1. |

### :SEARch:SHOLd:DLEVel

- **Command Format**

  :SEARch:SHOLd:DLEVel <level>

  :SEARch:SHOLd:DLEVel?

- **Functional Description**

  Set or query the threshold for the data source.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

  :SEARch:SHOLd:DLEVel 3 Set the threshold for the data source to 3 V.

  :SEARch:SHOLd:DLEVel? The query returns 3.000000e+00.

### :SEARch:SHOLd:SCL

- **Command Format**

  :SEARch:SHOLd:SCL {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}

  :SEARch:SHOLd:SCL?

- **Functional Description**

  Set the clock source for the setup & hold search.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2| CHANnel3| CHANnel4}.

- **For Example**

  :SEARch:SHOLd:SCL CHAN1 Set Channel 1 as the clock source.

  :SEARch:SHOLd:SCL? The query returns CHANnel1.

### :SEARch:SHOLd:CLEVel

- **Command Format**

  :SEARch:SHOLd:CLEVel <level>

  :SEARch:SHOLd:CLEVel?

- **Functional Description**

  Set or query the threshold for the clock source.

  <level>: Threshold value

- **Return Format**

  The query returns the threshold of the clock source in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

  :SEARch:SHOLd:CLEVel 3 Set the threshold for the clock source to 3 V.

  :SEARch:SHOLd:CLEVel? The query returns 3.000000e+00.

### :SEARch:SHOLd:POLarity

- **Command Format**

:SEARch:SHOLd:POLarity {POSitive|NEGative}

:SEARch:SHOLd:POLarity?

- **Functional Description**

Set the edge type for the setup & hold search to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns {POSitive|NEGative}.

- **For Example**

| | |
|---|---|
| :SEARch:SHOLd:POLarity POS | Set the edge type for the setup & hold search to POSitive (Rising edge). |
| :SEARch:SHOLd:POLarity? | The query returns POSitive. |

### :SEARch:SHOLd:PATTern

- **Command Format**

:SEARch:SHOLd:PATTern { HIGH | LOW }

:SEARch:SHOLd:PATTern?

- **Functional Description**

Set or query the data type for the setup & hold search to HIGH (High level), LOW (Low level).

- **Return Format**

The query returns { HIGH | LOW }.

- **For Example**

| | |
|---|---|
| :SEARch:SHOLd:PATTern HIGH | Set the data type for the setup & hold search to HIGH. |
| :SEARch:SHOLd:PATTern? | The query returns HIGH. |

### :SEARch:SHOLd:QUALifier

- **Command Format**

:SEARch:SHOLd:QUALifier { SETup | HOLD | SH }

:SEARch:SHOLd:QUALifier?

- **Functional Description**

Set the search condition to SETup (Setup time), HOLD (Hold time), or SH (Setup and Hold time).

- **Return Format**

The query returns { SETup | HOLD | SH }.

- **For Example**

| | |
|---|---|
| :SEARch:SHOLd:QUALifier HOLD | Set the search condition to HOLD. |
| :SEARch:SHOLd:QUALifier? | The query returns HOLD. |

### :SEARch:SHOLd:TIMe

- **Command Format**

  :SEARch:SHOLd:TIMe <time>

  :SEARch:SHOLd:TIMe?

- **Functional Description**

  Set the time interval for the setup & hold search.

- **Return Format**

  The query returns the current time interval, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:SHOLd:TIMe 1 | Set et the time interval for the setup & hold search to 1s. |
| :SEARch:SHOLd:TIMe? | The query returns 1.000000e+00. |

## Nth Edge Search

### :SEARch:NEDGe:SOURce

- **Command Format**

  :SEARch:NEDGe:SOURce <source>

  :SEARch:NEDGe:SOURce?

- **Functional Description**

  Set or query the search source.

  <source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

  CHANnel<n> indicates the physical channel, where n can be 1, 2, 3, or 4.

- **Return Format**

  The query returns the search source {CHANnel1|CHANnel2|CHANnel3|CHANnel4}.

- **For Example**

| | |
|---|---|
| :SEARch:NEDGe:SOURce CHANnel1 | Set the search source as Channel 1. |
| :SEARch:NEDGe:SOURce? | The query returns CHANnel1. |

### :SEARch:NEDGe:LEVel

- **Command Format**

  :SEARch:NEDGe:LEVel <level>

  :SEARch:NEDGe:LEVel?

- **Functional Description**

  Set or query the threshold.

<level>: Threshold value

- **Return Format**

The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:NEDGe:LEVel -3 | Set the threshold to -3 V. |
| :SEARch:NEDGe:LEVel? | The query returns -3.000000e+00. |

### :SEARch:NEDGe:POLarity

- **Command Format**

:SEARch:NEDGe:POLarity {POSitive|NEGative}

:SEARch:NEDGe:POLarity?

- **Functional Description**

Set the edge type for the Nth edge search to POSitive (Rising) or NEGative (Falling).

- **Return Format**

The query returns the edge type {POSitive|NEGative }.

- **For Example**

| | |
|---|---|
| :SEARch:NEDGe:POLarity POS | Set the edge type to POSitive (Rising edge). |
| :SEARch:NEDGe:POLarity? | The query returns POSitive. |

### :SEARch:NEDGe:TIMe

- **Command Format**

:SEARch:NEDGe:TIMe <time>

:SEARch:NEDGe:TIMe?

- **Functional Description**

Set the idle time for the Nth edge search.

- **Return Format**

The query returns the current idle time, with the unit in seconds (s).

- **For Example**

| | |
|---|---|
| :SEARch:NEDGe:TIMe 1 | Set the idle time for the Nth edge search to 1s. |
| :SEARch:NEDGe:TIMe? | The query returns 1.000000e+00. |

### :SEARch:NEDGe:EDGe

- **Command Format**

:SEARch:NEDGe:EDGe <value>

:SEARch:NEDGe:EDGe?

- **Functional Description**

Set the edge count for the Nth edge search.

<value>: Integer type, ranging from 1-65535.

- **Return Format**

The query returns the edge count of the Nth edge search.

- **For Example**

| | |
|---|---|
| :SEARch:NEDGe:EDGe 100 | Set the edge count to 100. |
| :SEARch:NEDGe:EDGe? | The query returns 100. |

## Code Pattern Search

### :SEARch:PATTern:LEVel

- **Command Format**

:SEARch:PATTern:LEVel <level>

:SEARch:PATTern:LEVel?

- **Functional Description**

Set or query the threshold.

<level>: Threshold value

- **Return Format**

The query returns the threshold in scientific notation. The unit conforms to the current amplitude unit.

- **For Example**

| | |
|---|---|
| :SEARch:PATTern:LEVel 3 | Set the threshold to 3 V. |
| :SEARch:PATTern:LEVel? | The query returns 3.000000e+00. |

### :SEARch:PATTern:PATTern

- **Command Format**

:SEARch:PATTern:PATTern <source>,<pch>

:SEARch:PATTern:PATTern? <source>

- **Functional Description**

Set or query the trigger code pattern for the specified source. X represents the default value.

<source>: {CHANnel1|CHANnel2|CHANnel3|CHANnel4|<Dx>}.

<Dx>: Digital channel {D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}.

<pch>: {H|L|X|R|F}.

- **Return Format**

  The query returns the current trigger code pattern of the specified source.

- **For Example**

| | |
|---|---|
| :SEARch:PATTern:PATTern CHANnel1,H | Set the code pattern of Channel 1 as H. |
| :SEARch:PATTern:PATTern? CHANnel1 | The query returns H. |

# NAVigate Command

This command is used to control the navigation function on the oscilloscope. This function is only available when the instrument is in the stop state.

### :NAVigate:ENABle

- **Command Format**

  :NAVigate:ENABle { {1|ON} | {0|OFF} }

  :NAVigate:ENABle?

- **Functional Description**

  Switch or query the navigation function to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :NAVigate:ENABle ON | Enable the navigation function. |
| :NAVigate:ENABle? | The query returns 1. |

### :NAVigate:MODe

- **Command Format**

  :NAVigate:MODe {TIMe|SEARch}

  :NAVigate:MODe?

- **Functional Description**

  Set or query the navigation mode.

  TIMe: Time navigation mode; SEARch: Search event navigation mode.

- **Return Format**

  The query returns {TIMe|SEARch}.

- **For Example**

| | |
|---|---|
| :NAVigate:MODe TIMe | Set the navigation mode to TIMe. |

:NAVigate:MODe? The query returns TIMe.

### :NAVigate:TIMe:SPEed

- **Command Format**

  :NAVigate:TIMe:SPEed {HIGH|NORMal|LOW}

  :NAVigate:TIMe:SPEed?

- **Functional Description**

  Set or query the waveform playback speed in the time navigation mode. HIGH represents fast speed, NORMal represents normal speed, LOW represents slow speed.

- **Return Format**

  The query returns {HIGH|NORMal|LOW}.

- **For Example**

  :NAVigate:TIMe:SPEed NORMal Set the waveform playback speed in the time navigation mode to NORMal.

  :NAVigate:TIMe:SPEed? The query returns NORMal.

### :NAVigate:PLAY

- **Command Format**

  :NAVigate:PLAY { {1|ON} | {0|OFF} }

  :NAVigate:PLAY?

- **Functional Description**

  Set or query whether the current navigation playback status is ON or OFF. This determines which the corresponding navigation mode is playback.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

  :NAVigate:PLAY ON Start playback of the waveform in time navigation mode.

  :NAVigate:PLAY? The query returns 1.

### :NAVigate:PLAY:MODe

- **Command Format**

  :NAVigate:PLAY:MODe {LOOP|SINGle}

  :NAVigate:PLAY:MODe?

- **Functional Description**

Set or query the navigation playback mode.

LOOP: Loop playback

SINGle: Single playback

- **Return Format**

The query returns {LOOP|SINGle}.

- **For Example**

| | |
|---|---|
| :NAVigate:PLAY:MODe LOOP | Set the navigation playback mode to LOOP. |
| :NAVigate:PLAY:MODe? | The query returns LOOP. |

### :NAVigate:PLAY:DIRection

- **Command Format**

:NAVigate:PLAY:DIRection {FORWard|BACKward}

:NAVigate:PLAY:DIRection?

- **Functional Description**

Set or query the navigation playback direction.

FORWard: Forward playback

BACKward: Backward playback

- **Return Format**

The query returns {FORWard|BACKward}.

- **For Example**

| | |
|---|---|
| :NAVigate:PLAY:DIRection FORWard | Set the navigation playback direction to FORWard. |
| :NAVigate:PLAY:DIRection? | The query returns FORWard. |

### :NAVigate:PLAY:NEXT

- **Command Format**

:NAVigate:PLAY:NEXT

- **Functional Description**

Set the navigation playback mode to shift right.

- **For Example**

| | |
|---|---|
| :NAVigate:PLAY:NEXT | Set the navigation playback mode to shift right. |

### :NAVigate:PLAY:BACK

- **Command Format**

:NAVigate:PLAY:BACK

- **Functional Description**

  Set the navigation playback mode to left right.

- **For Example**

| | |
|---|---|
| :NAVigate:PLAY:BACK | Set the navigation playback mode to left right. |

# MARKer Command

This command is used to control the oscilloscope's marker function.

### :MARKer:ENABle

- **Command Format**

  :MARKer:ENABle { {1|ON} | {0|OFF} }

  :MARKer:ENABle?

- **Functional Description**

  Set or query the marker state: ON or OFF.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :MARKer:ENABle ON | Enable marker function. |
| :MARKer:ENABle? | The query returns 1. |

### :MARKer:APPLy

- **Command Format**

  :MARKer:APPLy

- **Functional Description**

  Mark the waveform point.

- **For Example**

| | |
|---|---|
| :MARKer:APPLy | Mark the waveform point. |

### :MARKer:INDex

- **Command Format**

  :MARKer:INDex <index>

  :MARKer:INDex?

- **Functional Description**

  Set or query the waveform point marker index.

<index>: Index

- **Return Format**

The query returns the index as an integer.

- **For Example**

| | |
|---|---|
| :MARKer:INDex 3 | Set the waveform point marker index to 3. |
| :MARKer:INDex? | The query returns 3. |

### :MARKer:CLEar

- **Command Format**

:MARKer:CLEar

- **Functional Description**

Clear the waveform marker at the current time.

- **For Example**

| | |
|---|---|
| :MARKer:CLEar | Clear the waveform marker at the current time. |

### :MARKer:ALLClear

- **Command Format**

:MARKer:ALLClear

- **Functional Description**

Clear all waveform markers.

- **For Example**

| | |
|---|---|
| :MARKer:ALLClear | Clear all waveform markers. |

### :MARKer:FIRSt

- **Command Format**

:MARKer:FIRSt

- **Functional Description**

Select the first marker.

- **For Example**

| | |
|---|---|
| :MARKer:FIRSt | Select the first marker. |

### :MARKer:LAST

- **Command Format**

:MARKer:LAST

- **Functional Description**

  Select the last marker.

- **For Example**

| | |
|---|---|
| :MARKer:LAST | Select the last marker. |

### :MARKer:NEXT

- **Command Format**

  :MARKer:NEXT

- **Functional Description**

  Select the next marker on the right.

- **For Example**

| | |
|---|---|
| :MARKer:NEXT | Select the next marker on the right. |

### :MARKer:BACK

- **Command Format**

  :MARKer:BACK

- **Functional Description**

  Select the previous marker on the left.

- **For Example**

| | |
|---|---|
| :MARKer:BACK | Select the previous marker on the left. |

### :MARKer:DATA?

- **Command Format**

  :MARKer:DATA?

- **Functional Description**

  Query the marker result data.

- **Return Format**

  The query returns the marker result data. The returned data conforms to the Data Block Format.

- **For Example**

  MARKer:DATA?

  The query returns the marker result data.

  #9000000113MARKER

  Index,Time

  1,1.000000e+01

2,2.000000e+01

3,3.000000e+01

4,4.000000e+01

.....

# HISTogram Command

This command is used to control the zone histogram function on the oscilloscope.

### :HISTogram:ENABle

- **Command Format**

  :HISTogram:ENABle { {1|ON} | {0|OFF} }

  :HISTogram:ENABle?

- **Functional Description**

  Set or query the zone histogram function to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

| | |
|---|---|
| :HISTogram:ENABle ON | Enable the zone histogram function. |
| :HISTogram:ENABle? | The query returns 1. |

### :HISTogram:TYPe

- **Command Format**

  :HISTogram:TYPe {HORizontal|VERTical}

  :HISTogram:TYPe?

- **Functional Description**

  Set or query the zone histogram type.

  HORizontal: Horizontal histogram

  VERTical: Vertical histogram

- **Return Format**

  The query returns {HORizontal|VERTical}.

- **For Example**

| | |
|---|---|
| :HISTogram:TYPe HORizontal | Set the zone histogram type to HORizontal. |
| :HISTogram:TYPe? | The query returns HORizontal. |

### :HISTogram:SOURce

- **Command Format**

  :HISTogram:SOURce <source>

  :HISTogram:SOURce?

- **Functional Description**

  Set or query the source for the zone histogram type.

  <source>: CHANnel<n>, where n can take values 1, 2, 3, or 4.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4}.

- **For Example**

| | |
|---|---|
| :HISTogram:SOURce CHANnel1 | Set the source as Channel 1. |
| :HISTogram:SOURce? | The query returns CHANnel1. |

### :HISTogram:AREa

- **Command Format**

  :HISTogram:AREa <hp1>,<vp1>,<hp2>,<vp2>

  :HISTogram:AREa?

- **Functional Description**

  Set or query the histogram area, use the coordinates of the upper left and bottom right corners to define the screen range. The left boundary should be less than the right boundary, and the upper boundary should be greater than the lower boundary.

  <hp1>: Represents the horizontal time value of the upper left point of the area, with the unit in seconds (s).

  <vp1>: Represents the channel vertical value of the upper left point of the area. The unit is determined by the channel's vertical unit.

  <hp2>: Represents the horizontal time value of the bottom right point of the area, with the unit in seconds (s).

  <vp2>: Represents the channel vertical value of the bottom right point of the area. The unit is determined by the channel's vertical unit.

- **Return Format**

  The query returns the coordinate value in scientific notation.

- **For Example**

  :HISTogram:AREa -5us,200mv,5us,-200mv

  Zone A is from the upper left point [-5 us, 200 mv] to the bottom right point [5 us, -200 mv].

:HISTogram:AREa?

The query returns -5.000000e-06, 2.000000e-01, 5.000000e-06, -2.000000e-01.

### :HISTogram:STATistic:ENABle

- **Command Format**

  :HISTogram:STATistic:ENABle { {1|ON} | {0|OFF} }

  :HISTogram:STATistic:ENABle?

- **Functional Description**

  Set or query the state of zone histogram statistical function: ON or OFF.

- **Return Format**

  The query returns 1 or 0, indicating ON of OFF, respectively.

- **For Example**

| | |
|---|---|
| :HISTogram:STATistic:ENABle ON | Enable zone histogram statistical function. |
| :HISTogram:STATistic:ENABle? | The query returns 1. |

### :HISTogram:STATistic:RESet

- **Command Format**

  :HISTogram:STATistic:RESet

- **Functional Description**

  Clear the historical statistical data and restart the statistics.

- **For Example**

| | |
|---|---|
| :HISTogram:STATistic:RESet | Clear the historical statistical data and restart the statistics. |

### :HISTogram:STATistic:RESult?

- **Command Format**

  :HISTogram:STATistic:RESult?

- **Functional Description**

  Query the statistical results of the zone histogram. The returned data is determined by the command :HISTogram:TYPe and :HISTogram:SOURce.

- **Return Format**

  The query returns the statistical results arranged in CSV format in scientific notation. The returned data conforms to the Data Block Format.

- **For Example**

  :HISTogram:STATistic:RESult?

The query returns the statistical results:

#9000000122HISTOGRAM,

Sum,Peaks,Max,Min,Pk_Pk,Mean,Median,Mode,Width,Siqma

5000, 20, 4ms, -4ms,8ms,-20us,-20us,-4ms,20us,2.301ms

In which, “#9000000196” is the TMC data block header, followed by the data in the option list. In the data block header, the number following “#9” indicates the number of bytes of valid data that follows. HISTogram represents the histogram, with each piece of data separated by commas and each line of data separated by newline characters.

The statistical results include the following items.

Sum: Total count of statistical data.

Peaks: The maximum count of data being counted.

Max: The maximum value of the total statistical data.

Min: The minimum value of the total statistical data.

Pk_Pk: The difference between the maximum and minimum values (Max-Min) in the total statistical data.

Mean: The average of histogram.

Median: The median value of histogram.

Mode: The mode value of histogram.

Width: The width of histogram.

Sigma: The standard deviation of histogram.

# POWer Command

This command is used to control the power analysis function.

### :POWer:ENABle

- **Command Format**

  :POWer:ENABle { {1|ON} | {0|OFF} }

  :POWer:ENABle?

- **Functional Description**

  Set or query the power analysis function to ON or OFF.

- **Return Format**

  The query returns either 1 or 0, indicating ON or OFF, respectively.

- **For Example**

  :POWer:ENABle ON Enable the power analysis function.

| | |
|---|---|
| :POWer:ENABle? | The query returns 1. |

### :POWer:TYPe

- **Command Format**

  :POWer:TYPe {QUALity|HARMonics|INRush|RDS|SWITch|SLEW}

  :POWer:TYPe?

- **Functional Description**

  Set or query the power analysis type.

  QUALity: Power quality

  HARMonics: Current harmonics

  INRush: Surge current

  RDS: Dynamic on-state resistance analysis

  SWITch: Switch loss analysis

  SLEW: Conversion rate analysis.

- **Return Format**

  The query returns {QUALity|HARMonics|INRush|RDS|SWITch|SLEW}.

- **For Example**

| | |
|---|---|
| :POWer:TYPe QUALity | Set the power analysis type to QUALity. |
| :POWer:TYPe? | The query returns QUALity. |

### :POWer:VOLTage:SOURce

- **Command Format**

  :POWer:VOLTage:SOURce <source>

  :POWer:VOLTage:SOURce?

- **Functional Description**

  Set or query the input voltage source for power analysis.

  <source>: CHANnel<n>, where n can take values 1, 2, 3, or 4.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4}.

- **For Example**

| | |
|---|---|
| :POWer:VOLTage:SOURce CHANnel1 | Set the input voltage source as Channel 1. |
| :POWer:VOLTage:SOURce? | The query returns CHANnel1. |

### :POWer:CURRent:SOURce

- **Command Format**

  :POWer:CURRent:SOURce <source>

  :POWer:CURRent:SOURce?

- **Functional Description**

  Set or query the input current source for power analysis.

  <source>: CHANnel<n>, where n can take values 1, 2, 3, or 4.

- **Return Format**

  The query returns {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4}.

- **For Example**

| | |
|---|---|
| :POWer:CURRent:SOURce CHANnel1 | Set the input current source as Channel 1. |
| :POWer:CURRent:SOURce? | The query returns CHANnel1. |

### :POWer:QUALity:PERiod

- **Command Format**

  :POWer:QUALity:PERiod <count>

  :POWer:QUALity:PERiod?

- **Functional Description**

  Set or query the signal period count for power quality analysis.

  <count> represents period count, which is an integer.

- **Return Format**

  The query returns the period count as an integer.

- **For Example**

| | |
|---|---|
| :POWer:QUALity:PERiod 10 | Set the period count to 10. |
| :POWer:QUALity:PERiod? | The query returns 10. |

### :POWer:QUALity:APPLy

- **Command Format**

  :POWer:QUALity:APPLy

- **Functional Description**

  Execute the power quality statistical analysis.

- **For Example**

| | |
|---|---|
| :POWer:QUALity:APPLy | Execute the power quality statistical analysis. |

### :POWer:HARMonics:LFREQuency

- **Command Format**

  :POWer:HARMonics:LFREQuency {AUTO|50Hz|60Hz|400Hz}

  :POWer:HARMonics:LFREQuency?

- **Functional Description**

  Set or query the circuit frequency for the current harmonic analysis.

- **Return Format**

  The query returns {AUTO|50Hz|60Hz|400Hz}.

- **For Example**

| | |
|---|---|
| :POWer:HARMonics:LFREQuency 50Hz | Set the circuit frequency to 50 Hz. |
| :POWer:HARMonics:LFREQuency? | The query returns 50 Hz. |

### :POWer:HARMonics:IEC:CLAss

- **Command Format**

  :POWer:HARMonics:IEC:CLAss {A|B|C|D}

  :POWer:HARMonics:IEC:CLAss?

- **Functional Description**

  Set or query the standard for the current harmonic analysis.

  A: IEC61000-3-2 (Device standard for Class A).

  B: IEC61000-3-2 (Device standard for Class B).

  C: IEC61000-3-2 (Device standard for Class C).

  D: IEC61000-3-2 (Device standard for Class D).

- **Return Format**

  The query returns {A|B|C|D}.

- **For Example**

| | |
|---|---|
| :POWer:HARMonics:IEC:CLAss A | Set the analysis type to power quality. |
| :POWer:HARMonics:IEC:CLAss? | The query returns A. |

### :POWer:HARMonics:PERiod

- **Command Format**

  :POWer:HARMonics:PERiod <count>

  :POWer:HARMonics:PERiod?

- **Functional Description**

  Set or query the signal period count for the current harmonic analysis.

<count> represents period count as an integer.

- **Return Format**

  The query returns the period count as an integer.

- **For Example**

| | |
|---|---|
| :POWer:HARMonics:PERiod 10 | Set the period count to 10. |
| :POWer:HARMonics:PERiod? | The query returns 10. |

### :POWer:HARMonics:APPLy

- **Command Format**

  :POWer:HARMonics:APPLy

- **Functional Description**

  Execute the current harmonic statistical analysis.

- **For Example**

| | |
|---|---|
| :POWer:HARMonics:APPLy | Execute the current harmonic statistical analysis. |

### :POWer:HARMonics:FAIL?

- **Command Format**

  :POWer:HARMonics:FAIL?

- **Functional Description**

  Query the number of failures for the current harmonic statistical analysis.

- **Return Format**

  The query returns the number of failures as an integer.

- **For Example**

| | |
|---|---|
| :POWer:HARMonics:FAIL? | The query returns 10. |

### :POWer:HARMonics:STATus?

- **Command Format**

  :POWer:HARMonics:STATus?

- **Functional Description**

  Query the results status for the current harmonic statistical analysis.

  Result status: {PASS|FAIL|UNTested}, where PASS indicates that the test passed; FAIL indicates that the test is failed. UNTested indicates that the test has not been performed.

- **Return Format**

  The query returns the results status of the current harmonic statistical analysis.

- **For Example**

| | |
|---|---|
| :POWer:HARMonics:STATus? | The query returns FAIL. |

### :POWer:HARMonics:DATA?

- **Command Format**

:POWer:HARMonics:DATA?

- **Functional Description**

Query the result data for the current harmonic statistical analysis.

- **Return Format**

The query returns the statistical result data for the current harmonic analysis. The returned data conforms to Data Block Format.

- **For Example**

:POWer:HARMonics:DATA?

The query returns the statistical result data for the current harmonic analysis.

#9000000382HARMONICS,IEC61000-3-2 A,FAIL,3.000000e+03,1.000000e+03,

Harmonics,Value,Limits,Margin,Status

1,1.000000e+01,2.000000e+02,3.000000e+03,FAIL

2,1.000000e+01,2.000000e+02,3.000000e+03,FAIL

3,1.000000e+01,2.000000e+02,3.000000e+03,FAIL

4,1.000000e+01,2.000000e+02,3.000000e+03,PASS

.....

### :POWer:INRush:VOLTage

- **Command Format**

:POWer:INRush:VOLTage <voltage>

:POWer:INRush:VOLTage?

- **Functional Description**

Set or query the maximum input voltage for the surge current analysis.

<voltage>: Voltage value, with the unit in V.

- **Return Format**

The query returns the set voltage value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:INRush:VOLTage 5V | Set the maximum input voltage to 5 V. |
| :POWer:INRush:VOLTage? | The query returns 5.000000e+00 |

### :POWer:INRush:CURRent

- **Command Format**

  :POWer:INRush:CURRent <current>

  :POWer:INRush:CURRent?

- **Functional Description**

  Set or query the preset current for the surge current analysis.

  <current>: Current value, with the unit in A.

- **Return Format**

  The query returns the set current value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:INRush:CURRent 5A | Set the preset current to 5 A. |
| :POWer:INRush:CURRent? | The query returns 5.000000e+00. |

### :POWer:INRush:EXIT

- **Command Format**

  :POWer:INRush:EXIT

- **Functional Description**

  Exit the statistical analysis of the power surge current. This command is equivalent to pressing the Cancel soft key on the oscilloscope's front panel during the analysis.

- **For Example**

| | |
|---|---|
| :POWer:INRush:EXIT | Exit the statistical analysis of the power surge current. |

### :POWer:INRush:NEXT

- **Command Format**

  :POWer:INRush:NEXT

- **Functional Description**

  Execute the next statistical analysis of the power surge current. This command is equivalent to pressing the Next soft key on the oscilloscope's front panel during the analysis.

- **For Example**

| | |
|---|---|
| :POWer:INRush:NEXT | Execute the next statistical analysis of the power surge current. |

### :POWer:INRush:APPLy

- **Command Format**

:POWer:INRush:APPLy

- **Functional Description**

Execute the power surge current statistical analysis.

- **For Example**

| | |
|---|---|
| :POWer:INRush:APPLy | Execute the power surge current statistical analysis. |

### :POWer:RDS:APPLy

- **Command Format**

:POWer:RDS:APPLy

- **Functional Description**

Execute the dynamic on-resistance analysis.

- **For Example**

| | |
|---|---|
| :POWer:RDS:APPLy | Execute the dynamic on-resistance analysis. |

### :POWer:SWITch:CONDuction

- **Command Format**

:POWer:SWITch:CONDuction {WAVeform | RDS | VCE}

:POWer:SWITch:CONDuction?

- **Functional Description**

Set or query the conduction type and conductive calculation method for switching loss in power analysis.

WAVeform: The power waveform uses the original voltage waveform data, and power (P) is calculated as P = V × I.

RDS: For Rds(on), in the conduction region (when the voltage level is lower than VRef), the power is P = $Id^2$ × Rds (on); in the turn-off region (when the current level is lower than IRef), the power is P = 0 Watts.

VCE: For Vce (sat), in the conduction region (when the voltage level is lower than VRef), the power is P = Vce (sat) × Ic; in the turn-off region (when the current level is lower than IRef), the power is P = 0 Watts.

- **Return Format**

The query returns {WAVeform | RDS | VCE}

- **For Example**

| | |
|---|---|
| :POWer:SWITch:CONDuction RDS | Set the source reference as RDS. |
| :POWer:SWITch:CONDuction? | The query returns RDS. |

### :POWer:SWITch:IREFerence

- **Command Format**

  :POWer:SWITch:IREFerence <percent>

  :POWer:SWITch:IREFerence?

- **Functional Description**

  Set or query the current switching level at the start of the switching edge for power switching loss. This value is expressed as a percentage of the maximum switching current and can be adjusted to ignore the noise floor or zero offset that is difficult to eliminate in the current probe. It specifies the threshold used to determine the switching edge.

  <percent>: The percentage of the maximum switching current, expressed as an integer, with the unit in %.

- **Return Format**

  The query returns the current reference value.

- **For Example**

| | |
|---|---|
| :POWer:SWITch:IREFerence 10 | Set the current reference value to 10%. |
| :POWer:SWITch:IREFerence? | The query returns 10. |

### :POWer:SWITch:VREFerence

- **Command Format**

  :POWer:SWITch:VREFerence <percent>

  :POWer:SWITch:VREFerence?

- **Functional Description**

  Set or query the voltage switching level for the specified switching edge (voltage change point) in power switching loss. This value is expressed as a percentage of the maximum switching voltage and can be adjusted to account for the noise level. It determines the threshold used to identify when the signal switches (i.e., the rising edge or falling edge).

  <percent>: The percentage of the maximum switching voltage, expressed as an integer, with the unit in %.

- **Return Format**

  The query returns the voltage reference value.

- **For Example**

| | |
|---|---|
| :POWer:SWITch:VREFerence 10 | Set the voltage reference value to 10%. |
| :POWer:SWITch:VREFerence? | The query returns 10. |

### :POWer:SWITch:RDS

- **Command Format**

  :POWer:SWITch:RDS <value>

  :POWer:SWITch:RDS?

- **Functional Description**

  Set or query the Rds (on) value used in the RDS conduction calculation method for switching loss.

  <value>: Resistance value, with the unit in Ω.

- **Return Format**

  The query returns the resistance value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SWITch:RDS 10 | Set preset Rds (on) value to 10 Ω. |
| :POWer:SWITch:RDS? | The query returns 1.000000e+01. |

### :POWer:SWITch:VCE

- **Command Format**

  :POWer:SWITch:VCE <value>

  :POWer:SWITch:VCE?

- **Functional Description**

  Set or query the Vce (sat) value used in the VCE conduction calculation method for switching loss.

  <value>: Voltage value, with the unit in V.

- **Return Format**

  The query returns the set voltage value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SWITch:VCE 2 | Set preset Vce value to 2 V. |
| :POWer:SWITch:VCE? | The query returns 2.000000e+00. |

### :POWer:SWITch:APPLy

- **Command Format**

  :POWer:SWITch:APPLy

- **Functional Description**

  Execute the switching loss analysis.

- **For Example**

| | |
|---|---|
| :POWer:SWITch:APPLy | Execute the switching loss analysis. |

### :POWer:SLEW:POLarity

- **Command Format**

:POWer:SLEW:POLarity {POSitive|NEGative}

:POWer:SLEW:POLarity?

- **Functional Description**

Set or query the edge for the power analysis conversion rate. POSitive: Rising edge

NEGative: Falling edge

- **Return Format**

The query returns {POSitive|NEGative}.

- **For Example**

| | |
|---|---|
| :POWer:SLEW:POLarity POSitive | Set the edge to POSitive (Rising edge). |
| :POWer:SLEW:POLarity? | The query returns POSitive |

### :POWer:SLEW:APPLy

- **Command Format**

:POWer:SLEW:APPLy

- **Functional Description**

Execute switching rate analysis.

- **For Example**

| | |
|---|---|
| :POWer:SLEW:APPLy | Execute switching rate analysis. |

### :POWer:SOA:FAILed:OPERate

- **Command Format**

:POWer:SOA:FAILed:OPERate { NONE | STOP | SOUND }

:POWer:SOA:FAILed:OPERate?

- **Functional Description**

Set or query the failure operation for the safe operating area (SOA) in power analysis.

NONE: No operation

STOP: Stop the operation

SOUND: Triggers a sound prompt

- **Return Format**

The query returns { NONE | STOP | SOUND }.

- **For Example**

| | |
|---|---|
| :POWer:SOA:FAILed:OPERate STOP | Set the failure operation to STOP. |
| :POWer:SOA:FAILed:OPERate? | The query returns STOP. |

### :POWer:SOA:VOLTage:MIN

- **Command Format**

:POWer:SOA:VOLTage:MIN <voltage>

:POWer:SOA:VOLTage:MIN?

- **Functional Description**

Set or query the minimum voltage of the safe operating area.

<voltage>: Voltage value, with the unit in V.

- **Return Format**

The query returns the set voltage value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SOA:VOLTage:MIN 5V | Set the minimum voltage to 5 V. |
| :POWer:SOA:VOLTage:MIN? | The query returns 5.000000e+00. |

### :POWer:SOA:VOLTage:MAX

- **Command Format**

:POWer:SOA:VOLTage:MAX <voltage>

:POWer:SOA:VOLTage:MAX?

- **Functional Description**

Set or query the maximum voltage of the safe operating area.

<voltage>: Voltage value, with the unit in V.

- **Return Format**

The query returns the set voltage value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SOA:VOLTage:MAX 5V | Set the maximum voltage to 5 V. |
| :POWer:SOA:VOLTage:MAX? | The query returns 5.000000e+00. |

### :POWer:SOA:CURRent:MIN

- **Command Format**

:POWer:SOA:CURRent:MIN <current>

:POWer:SOA:CURRent:MIN?

- **Functional Description**

Set or query the minimum current of the safe operating area.

<current>: Current value, with the unit in A.

- **Return Format**

The query returns the set current value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SOA:CURRent:MIN 5A | Set the minimum current to 5 A. |
| :POWer:SOA:CURRent:MIN? | The query returns 5.000000e+00. |

### :POWer:SOA:CURRent:MAX

- **Command Format**

:POWer:SOA:CURRent:MAX <current>

:POWer:SOA:CURRent:MAX?

- **Functional Description**

Set or query the maximum current of the safe operating area.

<current>: Current value, with the unit in A.

- **Return Format**

The query returns the set current value in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SOA:CURRent:MAX 5A | Set the maximum current to 5 A. |
| :POWer:SOA:CURRent:MAX? | The query returns 5.000000e+00. |

### :POWer:SOA:MASK:TYPe

- **Command Format**

:POWer:SOA:MASK:TYPe { LIMit | CUSTom }

:POWer:SOA:MASK:TYPe?

- **Functional Description**

Set or query the template type for the safe operating area (SOA) in power analysis.

LIMit: Limit value

CUSTom: Customized value

- **Return Format**

The query returns { LIMit | CUSTom }.

- **For Example**

| | |
|---|---|
| :POWer:SOA:MASK:TYPe LIMit | Set the template type to LIMit. |
| :POWer:SOA:MASK:TYPe? | The query returns LIMit. |

### :POWer:SOA:LIMit:VOLTage

- **Command Format**

  :POWer:SOA:LIMit:VOLTage <voltage>

  :POWer:SOA:LIMit:VOLTage?

- **Functional Description**

  Set or query the voltage limit of the safe operating area.

  <voltage>: Voltage value, with the unit in V.

- **Return Format**

  The query returns the voltage limit in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SOA:LIMit:VOLTage 5V | Set the voltage limit to 5 V. |
| :POWer:SOA:LIMit:VOLTage? | The query returns 5.000000e+00. |

### :POWer:SOA:LIMit:CURRent

- **Command Format**

  :POWer:SOA:LIMit:CURRent <current>

  :POWer:SOA:LIMit:CURRent?

- **Functional Description**

  Set or query the current limit of the safe operating area.

  <current>: Current value, with the unit in A.

- **Return Format**

  The query returns the current limit in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SOA:LIMit:CURRent 5A | Set the current limit to 5 A. |
| :POWer:SOA:LIMit:CURRent? | The query returns 5.000000e+00. |

### :POWer:SOA:LIMit:POWer

- **Command Format**

  :POWer:SOA:LIMit:POWer <power>

  :POWer:SOA:LIMit:POWer?

- **Functional Description**

Set or query the power limit of the safe operating area.

<power>: Power value, with the unit in W.

- **Return Format**

The query returns the power limit in scientific notation.

- **For Example**

| | |
|---|---|
| :POWer:SOA:LIMit:POWer 25W | Set the power limit to 25 W. |
| :POWer:SOA:LIMit:POWer? | The query returns 2.5000000e+01. |

### :POWer:SOA:CUSTom:CREate

- **Command Format**

:POWer:SOA:CUSTom:CREate <data>

:POWer:SOA:CUSTom:CREate?

- **Functional Description**

Create or query customized template data in safety operation area.

<data>: The query returns a binary data block. The list data is arranged in the CSV format, and the returned data conforms to the Data Block Format. When sending, the configured data format must match the actual data format of the query. Attach the binary data stream to the command string and complete the transmission in a single operation.

- **Return Format**

The query returns customized template data. The list data is arranged in the CSV format, and the returned data conforms to the Data Block Format.

- **For Example**

Create customized template data.

:POWer:SOA:CUSTom:CREate #9000000727<data>

The query returns the customized template data.

:POWer:SOA:CUSTom:CREate?

#9000000727

Points,X(V),Y(A),

1,0.000000e+00,1.000000e+01,

2,1.000000e+01,1.000000e+01,

3,5.000000e+00,0.000000e+00

.....

### :POWer:SOA:CUSTom:LOAD

- **Command Format**

  :POWer:SOA:CUSTom:LOAD <filepath>

- **Functional Description**

  Load the specified rule file as a customized template.

  <filepath> indicates the file absolute path and must be a character string enclosed in double quotation marks.

  Note: In the path string, Local:/ refers to the internal instrument storage, while Udisk:/ refers to the external USB flash drive.

- **For Example**

| | |
|---|---|
| :POWer:SOA:CUSTom:LOAD "Local:/wave/test.csv" | Load test.csv rule file from the local storage. |
| :POWer:SOA:CUSTom:LOAD "Udisk:/wave/test.csv" | Load test.csv rule file from USB flash drive. |

### :POWer:SOA:CUSTom:SAVe

- **Command Format**

  :POWer:SOA:CUSTom:SAVe <filepath>

- **Functional Description**

  Save the current setting parameters to a customized template file.

  <filepath> indicates the file absolute path and must be a character string enclosed in double quotation marks.

  Note: In the path string, Local:/ refers to the internal instrument storage, while Udisk:/ refers to the external USB flash drive.

- **For Example**

| | |
|---|---|
| :POWer:SOA:CUSTom:SAVe "Local:/wave/test.csv" | Save the test.csv file to the local storage. |
| :POWer:SOA:CUSTom:SAVe "Udisk:/wave/test.csv" | Save the test.csv file to the USB flash driver. |

### :POWer:SOA:RESet

- **Command Format**

  :POWer:SOA:RESet

- **Functional Description**

Execute a reset for the safe operating area analysis.

- **For Example**

:POWer:SOA:RESet Execute a reset for the safe operating area analysis.

### :POWer:SOA:APPLy

- **Command Format**

:POWer:SOA:APPLy

- **Functional Description**

Execute the safe operating area analysis.

- **For Example**

:POWer:SOA:APPLy Execute the safe operating area analysis.

### :POWer:SOA:DATA?

- **Command Format**

:POWer:SOA:DATA?

- **Functional Description**

Query the current statistical result data of the safe operating area analysis.

The returned data format: <pass>, <failed>, <total>, <result>. Here, <pass> represents the number of passes, <failed> represents the number of failures, <total> represents the total count, and <result> represents the outcome, where 1 indicates success and 0 indicates failure.

- **Return Format**

The query returns the statistical result data of the safe operating area analysis.

- **For Example**

:POWer:SOA:DATA? The query returns the statistical result data of the safe operating area analysis 138,0,138,1.

### :POWer:MODulation:REFSource

- **Command Format**

:POWer:MODulation:REFSource { V | I }

:POWer:MODulation:REFSource?

- **Functional Description**

Set or query the source reference for modulation analysis in power analysis.

V: Voltage

I: Current

- **Return Format**

  The query returns { V | I }.

- **For Example**

| | |
|---|---|
| :POWer:MODulation:REFSource V | Set the source reference as V (Voltage). |
| :POWer:MODulation:REFSource? | The query returns V. |

### :POWer:MODulation:TYPe

- **Command Format**

  :POWer:MODulation:TYPe {PERiod|FREQuency|RTIMe|FTIMe|PWIDth|NWIDth|PDUTy|NDUTy}

  :POWer:MODulation:TYPe?

- **Functional Description**

  Set or query the parameter analysis for modulation analysis.

  PERiod: Period

  FREQuency: Frequency

  RTIMe: Rise Time

  FTIMe: Fall Time

  PWIDth: Positive Pulse Width

  NWIDth: Negative Pulse Width

  PDUTy: Positive Duty Cycle

  NDUTy: Negative Duty Cycle

- **Return Format**

  The query returns {PERiod|FREQuency|RTIMe|FTIMe|PWIDth|NWIDth|PDUTy|NDUTy}.

- **For Example**

| | |
|---|---|
| :POWer:MODulation:TYPe PERiod | Set the parameter analysis for modulation analysis to PERiod. |
| :POWer:MODulation:TYPe? | The query returns PERiod. |

### :POWer:MODulation:APPLy

- **Command Format**

  :POWer:MODulation:APPLy

- **Functional Description**

  Execute modulation analysis.

- **For Example**

| | |
|---|---|
| :POWer:MODulation:APPLy | Execute modulation analysis. |

### :POWer:MODulation:DATA?

- **Command Format**

  :POWer:MODulation:DATA?

- **Functional Description**

  Query the statistical result data corresponding to the current modulation analysis type.

  The return data format: <min>, <max>, <average>, where <min> represents the minimum value, <max> represents the maximum value, and <average> represents the average value.

- **Return Format**

  The query returns the statistical result data of the modulation analysis, expressed in scientific notation. The unit depends on the modulation analysis type.

- **For Example**

  :POWer:MODulation:DATA?

  The query returns the statistical result data 9.998004e+02,1.001001e+03,1.000200e+03

### :POWer:ITEM?

- **Command Format**

  :POWer:ITEM? <item>

- **Functional Description**

  Obtain the statistical results of the corresponding measurement items for power quality analysis. The measurement-related settings also apply to this function.

  <item> represents the measurement items:

  {VCRest|VRMS|VCRFactor|ICRest|IRMS|ICRFactor|TPWR|RPWR|APWR|PWRFactor|PHASe|INRush|RDSon|PLOSs|ELOSs|SRMAX|SRMIN} represents voltage peak, RMS voltage, voltage peak factor, current peak, RMS current, current peak factor, active power, reactive power, apparent power, power factor, phase angle, surge current, dynamic on-state resistance, power loss, energy loss, maximum conversion rate, and minimum conversion rate, respectively.

- **Return Format**

  The query returns the statistical results with standard units in scientific notation.

- **For Example**

  :POWer:ITEM? VRMS

  The query returns the maximum statistical value of RMS voltage 1.120000e+00 under power quality.

### :POWer:STATistic:ITEM?

- **Command Format**

  :POWer:STATistic:ITEM? <type>,<item>

- **Functional Description**

  Obtain the statistical results of the corresponding measurement types for power quality analysis. The measurement-related settings also apply to this function.

  <item> represents the measurement items: {VCRest|VRMS|VCRFactor|ICRest|IRMS|ICRFactor|TPWR|RPWR|APWR|PWRFactor|PHASe|INRush|RDSon|PLOSs|ELOSs|SRMAX|SRMIN} represents voltage peak, RMS voltage, voltage peak factor, current peak, RMS current, current peak factor, active power, reactive power, apparent power, power factor, phase angle, surge current, dynamic on-state resistance, power loss, energy loss, maximum conversion rate, and minimum conversion rate, respectively.

  <type> represents statistical type: {MAXimum|MINimum|CURRent|AVERages|DEViation} represents maximum, minimum, current value, average, and variance, respectively.

- **Return Format**

  The query returns the statistical results with standard units in scientific notation.

- **For Example**

  :POWer:STATistic:ITEM? MAX,VRMS

  The query returns the maximum statistical value of RMS voltage 1.120000e+00 under power quality.

### :POWer:STATistic:HISTogram:RESult?

- **Command Format**

  :POWer:STATistic:HISTogram:RESult? <item>

- **Functional Description**

  Obtain the histogram statistical results of the corresponding measurement items for power quality analysis, sorted by the left boundary value, right boundary value, and probability percentage. The measurement-related settings also apply to this function.

  <item> represents the measurement items: {VCRest|VRMS|VCRFactor|ICRest|IRMS|ICRFactor|TPWR|RPWR|APWR|PWRFactor|PHASe|INRush|RDSon|PLOSs|ELOSs|SRMAX|SRMIN} represents voltage peak, RMS voltage, voltage peak factor, current peak, RMS current, current peak factor, active power, reactive power, apparent power, power factor, phase angle, surge current, dynamic on-state resistance, power loss, energy loss, maximum conversion rate, and minimum conversion rate, respectively.

## Return Format

The query returns the histogram statistical results arranged in CSV format in scientific notation.

The returned data conforms to the Data Block Format.

## For Example

:POWer:STATistic:HISTogram:RESult? VRMS

The query returns the histogram of statistical results for the current power analysis measurement:

#9000000128HISTOGRAM,

Sum,Peaks,Max,Min,Pk_Pk,Mean,Median,Mode, Siqma

100, 93, -8.000mV, -16.000mV,8.000mV,-8.400mV,-8.000mV,-8.000mV, 2.400mV

In which, “#9000000148” is the TMC data block header, followed by the data in the option list. In the data block header, the number following “#9” indicates the number of bytes of valid data that follows. HISTogram represents the histogram, with each piece of data separated by commas and each line of data separated by newline characters.

The statistical results include the following items.

Sum: Total count of statistical data.

Peaks: The maximum count of data being counted.

Max: The maximum value of the total statistical data.

Min: The minimum value of the total statistical data.

Pk_Pk: The difference between the maximum and minimum values (Max-Min) in the total statistical data.

Mean: The average of histogram.

Median: The median value of histogram.

Mode: The mode value of histogram.

Width: The width of histogram.

Sigma: The standard deviation of histogram.

# KEY Command

This command is used to control the keys and rotary knobs on the operation panel of the oscilloscope.

### Key List

| Key | Functional Description | LED |
|---|---|---|
| C1 | Channel 1 switch | √ |
| CH2 | Channel 2 switch | √ |
| CH3 | Channel 3 switch | √ |
| CH4 | Channel 4 switch | √ |
| MATH | Mathematical operation and menu | √ |
| AUTO | Automatic settings for displaying the appropriate waveform. | |
| RS | Control the running state. The oscilloscope will alternate between stop and running states if this command is sent continuously. | √ |
| SINGle | Single trigger | √ |
| CLEar | Clear | |
| TMENu | Trigger menu | |
| HMENu | Horizontal system menu | |
| MEASure | Measurement function | |
| CURSor | Cursor measurement and menu | |
| ACQuire | Acquisition menu | |
| DISPlay | Display menu | |
| STORage | Storage menu | |
| UTILity | Auxiliary menu | |
| APP | Main function menu | |
| BUS | Bus menu | √ |
| GEN | Waveform generation menu | √ |
| DEFault | Reset to default | |
| PSCReen | One-key save screen image | |
| REF | Reference waveform menu | √ |
| DIGital | Digital channel menu | √ |
| FFT | Spectrum menu | √ |
| TLOCk | Touch lock/unlock | √ |
| LEFT | Left functional key | |
| RIGHt | Right functional key | |

| | | |
|---|---|---|
| TFORe | Force trigger | |
| MODE | Mode switching key | |
| FKNob | Multi-function rotary knob | |
| FKNLeft | Multi-function left rotary knob | |
| FKNRight | Multi-function right rotary knob | |
| VPKNob | Vertical rotary knob | |
| VPKNLeft | Vertical left rotary knob | |
| VPKNRight | Vertical right rotary knob | |
| HPKNob | Horizontal rotary knob | |
| HPKNLeft | Horizontal left rotary knob | |
| HPKNRight | Horizontal right rotary knob | |
| TPKNob | Trigger rotary knob | |
| TPKNLeft | Trigger left rotary knob | |
| TPKNRight | Trigger right rotary knob | |
| VBKNob | Voltage reference rotary knob | |
| VBKNLeft | Voltage reference left rotary knob | |
| VBKNRight | Voltage reference right rotary knob | |
| TBKNob | Time reference rotary knob | |
| TBKNLeft | Time reference left rotary knob | |
| TBKNRight | Time reference right rotary knob | |
| ASTAtus | Automatic state indicator: no key, only light | √ |
| NSTAtus | Normal state indicator: no key, only light | √ |
| SSTAtus | Single indicator: no key, only light | √ |
| FSTAtus | Multi-function rotary knob indicator: no key, only light | √ |
| TBSTAtus | Time reference rotary knob indicator: no key, only light | √ |
| HPSTAtus | Horizontal rotary knob indicator: no key, only light | √ |
| TPSTAtus | Trigger rotary knob indicator: no key, only light | √ |
| VPSTAtus | Vertical rotary knob indicator: no key, only light | √ |
| VBSTAtus | Voltage reference rotary knob indicator: no key, only light | √ |

## :KEY:<key>

### ■ Command Format

:KEY:<key>

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

:KEY:<key>:LOCK?

:KEY:<key>:LED?

## Functional Description

Set the key function and its lock/unlock state, refer to Key List for <key> definition and description.

## Return Format

The query returns the key lock state or LED state.

Lock state: 0 indicates that the key is unlocked, while 1 indicates that the key is locked.

LED state: 0 indicates that LED is off, while 1 indicates that the key is on. For the RUN/STOP key, 0 indicates red and 1 indicates green.

## For Example

| | |
|---|---|
| :KEY:C1 | Press Channel 1 key. |
| :KEY:C1:LOCK ON/OFF | Lock/unlock Channel 1 key. |
| :KEY:C1:LOCK? | The query returns Channel 1 key state, 1 indicates that this key is locked. |
| :KEY:C1:LED? | he query returns Channel 1 LED state, 0 indicates that LED is off. |

# Programming Explanation

This chapter describes troubleshooting during the programming process. If you encounter any of the following problems, please follow the related instructions.

## Programming Preparation

The programming preparation is only applicable for using Visual Studio and LabVIEW development tools for programming under the Windows operating system.
Firstly, the user needs to confirm whether the NI-VISA library is installed (it can be download from the website https://www.ni.com/en-ca/support/downloads/drivers/download.ni-visa.html). In this manual, the default installment path is “C:\Program Files\IVI Foundation\VISA."

Establish communication with a PC via the USB or LAN interface of the instrument. Use a USB data cable to connect the USB DEVICE port on the rear panel of the instrument to the USB port of the PC, or use a LAN data cable to connect the LAN port on the rear panel of the instrument to the LAN port of the PC.

# VISA Programming Example

This section contains examples. Through these examples, users can learn how to use VISA and combine it with the commands in the programming manual to control the instrument. With these examples, users can develop more applications.

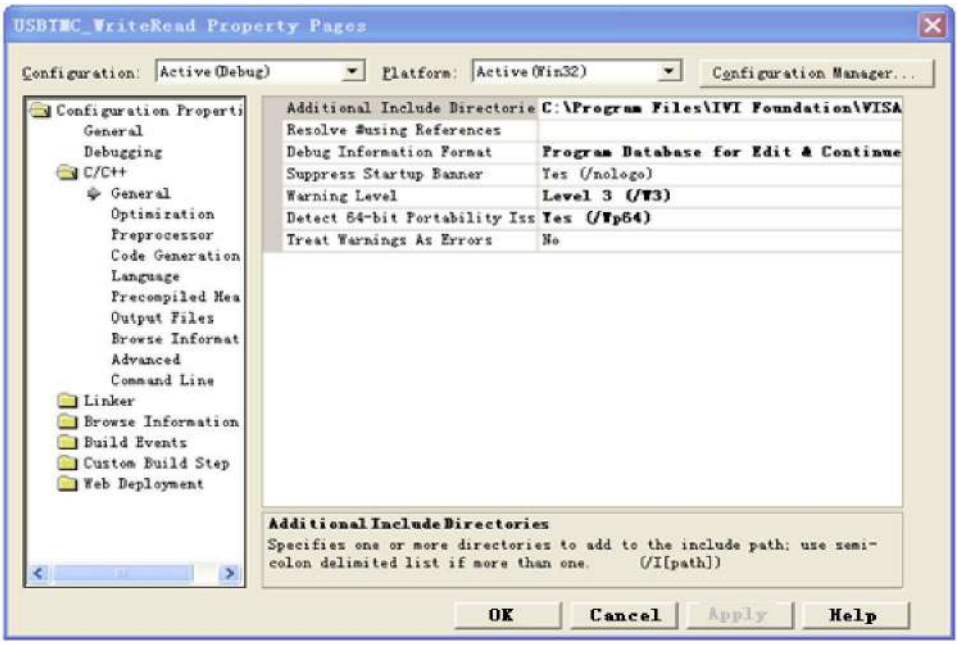
## VC++ Example

- Environment: Window System, Visual Studio.
- Description: Access the instrument via USBTMC and TCP/IP, and send “*IDN?” command on NI-VISA to query the device information.
- Steps:

1. Open the visual studio software to create a new VC++ win32 console project.
2. Set the project environment that can adjust NI-VISA library, which includes both static and dynamic libraries.
   a) Static library:
   Locate the files “visa.h, visatype.h, and visa32.lib” in the NI-VISA installation path. Copy them to the root directory of the VC++ project and add them to the project. Add the following two lines of code to the projectname.cpp file:

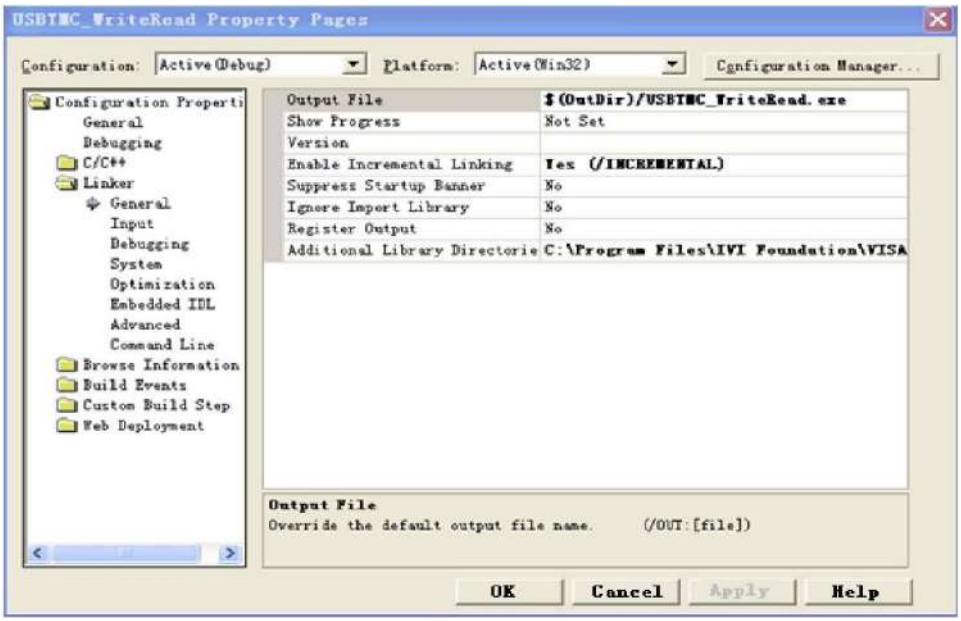
```
#include "visa.h"
#pragma comment (lib,"visa32.lib")
```

   b) Dynamic library:
   Press “project>>properties”, select "c/c++---General" in the attribute dialog on the leftside, and set the value of "Additional Include Directories" to the installment path of NI-VIS (for example, C:\ProgramFiles\IVI Foundation\VISA\WinNT\include), as shown in the following figure.

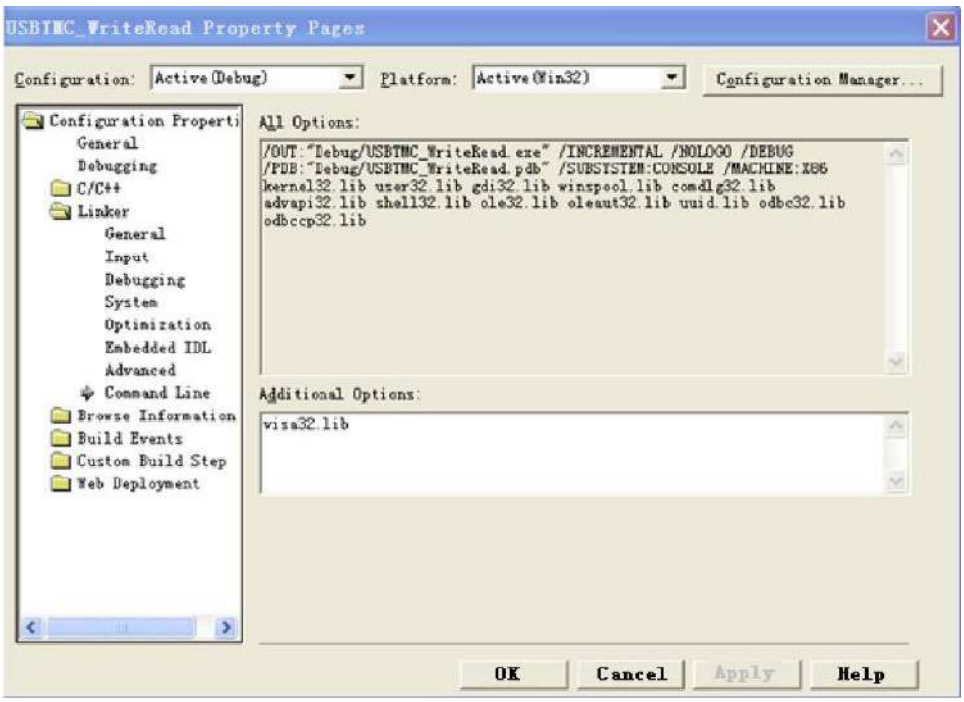



Select "Linker-General" in the attribute dialog on the left side, and set the value of "Additional Library Directories" as the installment path of NI-VIS

(for example, C:\Program Files\IVI Foundation\VISA\WinNT\include), as shown in the following figure.




Select "Linker-Command Line" in the attribute dialog on the left side, and set the value of "Additional" as visa32.lib, as shown in the following figure.




Add the file visa.h in the projectname.cpp file.

#include <visa.h>

## 1. Source code:

a) USBTMC Example

```
int usbtmc_test()
{ /** This code demonstrates sending synchronous read & write commands
    * to an USB Test & Measurement Class (USBTMC) instrument using NI-VISA
* The example writes the "*IDN?\n" string to all the USBTMC
* devices connected to the system and attempts Read back
* results using the write and read functions.
* Open Resource Manager
* Open VISA Session to an Instrument
* Write the Identification Query Using viPrintf
* Try Read a Response With viScanf
* Close the VISA Session*/
ViSession defaultRM;
ViSession instr;
ViUInt32 numInstrs;
ViFindList findList;
ViStatus status;
char instrResourceString[VI_FIND_BUFLEN];
unsigned char buffer[100];
int i;
```

```
status = viOpenDefaultRM(&defaultRM);
if (status < VI_SUCCESS)
{
    printf("Could not open a session to the VISA Resource Manager!\n");
    return status;
}
  /*Find all the USB TMC VISA resources in our system and store the number of resources in the system in
  numInstrs.*/
status = viFindRsrc(defaultRM, "USB?*INSTR", &findList, &numInstrs, instrResourceString);
if (status<VI_SUCCESS)
{
    printf("An error occurred while finding resources. \nPress Enter to continue.");
    fflush(stdin);
    getchar();
    viClose(defaultRM);
    return status;
}
/** Now we will open VISA sessions to all USB TMC instruments.
*   We must use the handle from viOpenDefaultRM and we must
*   also use a string that indicates which instrument Open. This
*   is called the instrument descriptor. The format for this string
*   can be found in the function panel by right clicking on the
*   descriptor parameter. After opening a session to the
*   device, we will get a handle to the instrument which we
*   will use in later VISA functions. The AccessMode and Timeout
*   parameters in this function are reserved for future
*   functionality. These two parameters are given the value VI_NULL. */
for (i = 0; i < int(numInstrs); i++)
{
    if (i > 0)
    {
        viFindNext(findList, instrResourceString);
    }
    status = viOpen(defaultRM, instrResourceString, VI_NULL, VI_NULL, &instr);
    if (status < VI_SUCCESS)
    {
        printf("Cannot open a session to the device %d. \n", i + 1);
        continue;
    }
```

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

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

b) TCP/IP Example

```
int tcp_ip_test(char *pIP)
{
char outputBuffer[VI_FIND_BUFLEN];
ViSession defaultRM, instr;
ViStatus status;
/* First we will need Open the default resource manager. */
status = viOpenDefaultRM(&defaultRM);
if (status < VI_SUCCESS)
{
    printf("Could not open a session to the VISA Resource Manager!\n");
}
/* Now we will open a session via TCP/IP device */
char head[256] = "TCPIP0::";
char tail[] = "::inst0::INSTR";
strcat(head, pIP);
strcat(head, tail);
status = viOpen(defaultRM, head, VI_LOAD_CONFIG, VI_NULL, &instr);
if (status < VI_SUCCESS)
{
    printf("An error occurred opening the session\n");
    viClose(defaultRM);
}
status = viPrintf(instr, "*idn?\n");
status = viScanf(instr, "%t", outputBuffer);
if (status < VI_SUCCESS)
{
    printf("viRead failed with error code: %x \n", status);
    viClose(defaultRM);
}
else
{
    printf("\nMesseage read from device: %*s\n", 0, outputBuffer);
}
status = viClose(instr);
status = viClose(defaultRM);
printf("Press Enter to exit.");
fflush(stdin);
getchar();
```

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

# C# Example

- Environment: Window System, Visual Studio.
- Description: Access the instrument via USBTMC and TCP/IP, and send "*IDN?" command on NI-VISA to query the device information.
- Steps:

1. Open the Visual Studio software to create a new C# console project.
2. Add C#, quote Ivi.Visa.dll, and NationalInstruments.Visa.dll of VISA.
3. Source code:

a) USBTMC Example

```
class Program
{
    void usbtmc_test()
    {
        using (var rmSession = new ResourceManager())
        {
            var resources = rmSession.Find("USB?*INSTR");
            foreach (string s in resources)
            {
                try
                {
                    var mbSession = (MessageBasedSession)rmSession.Open(s);
                    mbSession.RawIO.Write("*IDN?\n");
                    System.Console.WriteLine(mbSession.RawIO.ReadString());
                }
                catch (Exception ex)
```

```
                    {
                        System.Console.WriteLine(ex.Message);
                    }
                }
            }
        }

        void Main(string[] args)
        {
            usbtmc_test();
        }
    }
```

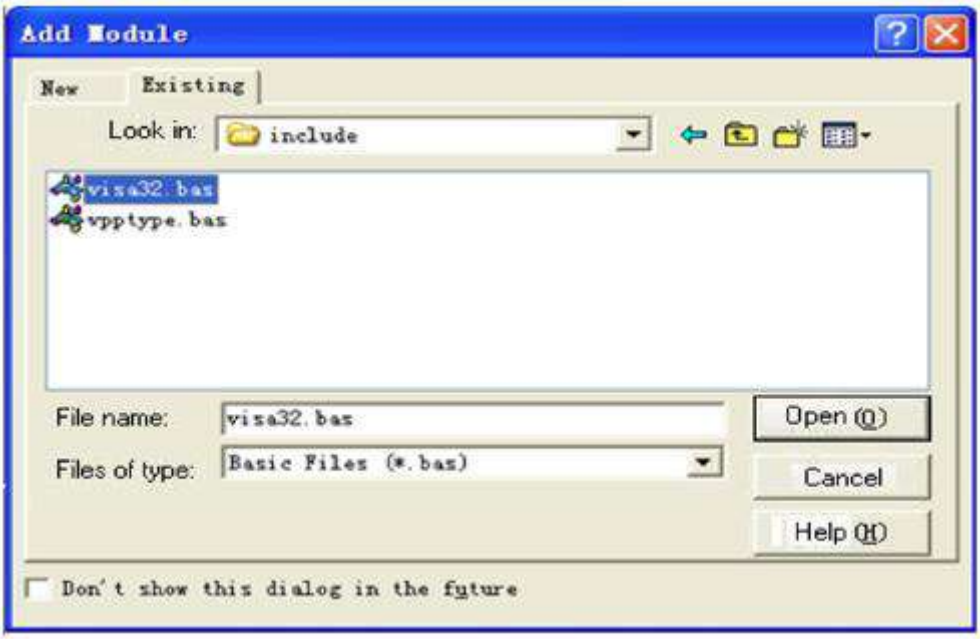
b) TCP/IP Example

```
    class Program
     {
        void tcp_ip_test(string ip)
        {
            using (var rmSession = new ResourceManager())
            {
                try
                {
                    var resource = string.Format("TCPIP0::{0}::inst0::INSTR", ip);
                    var mbSession = (MessageBasedSession)rmSession.Open(resource);
                    mbSession.RawIO.Write("*IDN?\n");
                    System.Console.WriteLine(mbSession.RawIO.ReadString());
                }
                catch (Exception ex)
                {
                    System.Console.WriteLine(ex.Message);
                }
            }
        }

        void Main(string[] args)
        {
            tcp_ip_test("192.168.20.11");
        }
     }
```

# VB Example

- Environment: Window System, Microsoft Visual Basic 6.0.
- Description: Access the instrument via USBTMC and TCP/IP, and send “*IDN?” command on NI-VISA to query the device information.
- Steps:

1. Open the Visual Basic software and create a new standard application program project.
2. Set the project environment to include the NI-VISA library. Press the “Existing” tab in Project >> Add Existing Item, locate the “visa32.bas” file in the “include” folder under the NI-VISA installation path, and add it, as shown below.




3. Source code:

a) USBTMC Example

```
PrivateFunction usbtmc_test() AsLong
' This code demonstrates sending synchronous read & write commands
' to an USB Test & Measurement Class (USBTMC) instrument using NI-VISA
' The example writes the "*IDN?\n" string to all the USBTMC
' devices connected to the system and attempts Read back
' results using the write and read functions.
' The general flow of the code is
' Open Resource Manager
' Open VISA Session to an Instrument
' Write the Identification Query Using viWrite
' Try Read a Response With viRead
' Close the VISA Session
```

```
Const MAX_CNT = 200
Dim defaultRM AsLong
Dim instrsesn AsLong
Dim numInstrs AsLong
Dim findList AsLong
Dim retCount AsLong
Dim status AsLong
Dim instrResourceString AsString *VI_FIND_BUFLEN
Dim Buffer AsString * MAX_CNT
Dim i AsInteger

' First we must call viOpenDefaultRM to get the manager
' handle. We will store this handle in defaultRM.
status = viOpenDefaultRM(defaultRM)
If(status < VI_SUCCESS) Then
    resultTxt.Text = "Could not open a session to the VISA Resource Manager!"
    usbtmc_test = status
ExitFunction
EndIf

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

' Now we will open VISA sessions to all USB TMC instruments.
' We must use the handle from viOpenDefaultRM and we must
' also use a string that indicates which instrument Open.   This
' is called the instrument descriptor.   The format for this string
' can be found in the function panel by right clicking on the
' descriptor parameter. After opening a session to the
' device, we will get a handle to the instrument which we
' will use in later VISA functions.   The AccessMode and Timeout
' parameters in this function are reserved for future
' functionality.   These two parameters are given the value VI_NULL.
```

```
For i = 0 To numInstrs
If (i > 0) Then
    status = viFindNext(findList, instrResourceString)
EndIf
    status = viOpen(defaultRM, instrResourceString, VI_NULL, VI_NULL, instrsesn)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Cannot open a session to the device " + CStr(i + 1)
GoTo NextFind
EndIf

' At this point we now have a session open to the USB TMC instrument.
' We will now use the viWrite function to send the device the string "*IDN?",
' asking for the device's identification.
status = viWrite(instrsesn, "*IDN?", 5, retCount)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error writing to the device."
    status = viClose(instrsesn)
GoTo NextFind
EndIf

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

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

b) TCP/IP Example

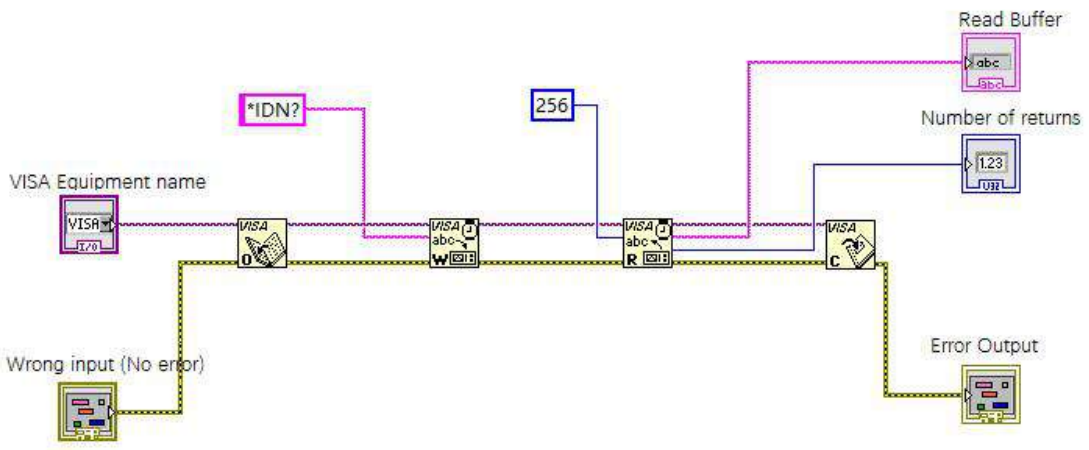
```
PrivateFunction tcp_ip_test(ByVal ip AsString) AsLong
```

```
Dim outputBuffer AsString * VI_FIND_BUFLEN
Dim defaultRM AsLong
Dim instrsesn AsLong
Dim status AsLong
Dim count AsLong

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

' Now we will open a session via TCP/IP device
status = viOpen(defaultRM, "TCPIP0::" + ip + "::inst0::INSTR", VI_LOAD_CONFIG, VI_NULL, instrsesn)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "An error occurred opening the session"
    viClose(defaultRM)
    tcp_ip_test = status
ExitFunction
EndIf
status = viWrite(instrsesn, "*IDN?", 5, count)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error writing to the device."
EndIf
    status = viRead(instrsesn, outputBuffer, VI_FIND_BUFLEN, count)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error reading a response from the device." + CStr(i + 1)
Else
    resultTxt.Text = "read from device:" + outputBuffer
EndIf
    status = viClose(instrsesn)
    status = viClose(defaultRM)
    tcp_ip_test = 0
EndFunction
```

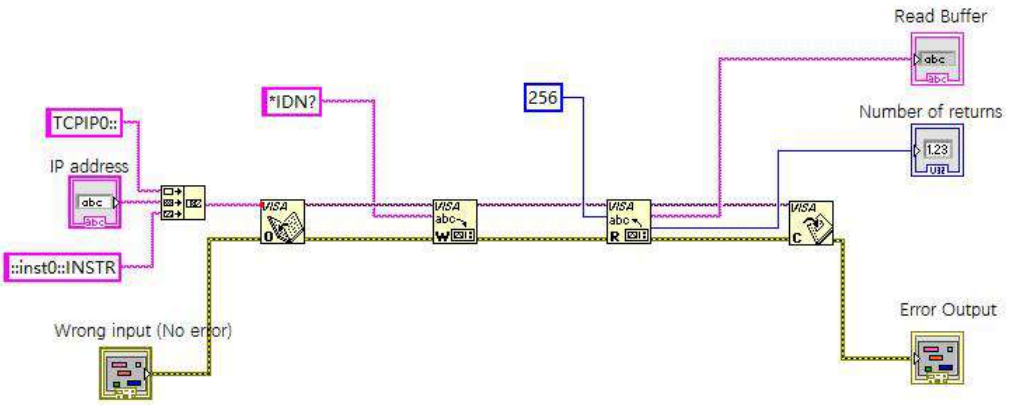
# LabVIEW Example

- Environment: Window System, LabVIEW.
- Description: Access the instrument via USBTMC and TCP/IP, and send "*IDN?" command on NI-VISA to query the device information.
- Steps:

1. Open the LabVIEW software to create a VI file.
2. Add the control, click the front panel Select and add the VISA source name, error input, error output, and part of the indicator from the control column.
3. Open the diagram, click the VISA source name, and then select and add these functions VISA Write, VISA Read, VISA Open, and VISA Close on the VISA menu.
4. The VI opens a VISA session for a USBTMC device, writes the *IDN? command to the device, and reads back the response value. When all communication is complete, the VI closes the VISA session, as shown in the following figure.




5. Communication with the device via TCP/IP is similar to USBTMC. You need Set the VISA Write and Read functions to synchronous I/O, as LabVIEW uses asynchronous I/O by default. Right-click on the node and select “Synchronous I/O Mode >> Synchronous” from the shortcut menu to enable synchronous writing or reading of data, as shown in the following figure.




# MATLAB Example

- Environment: Window System, MATLAB.
- Description: Access the instrument via USBTMC and TCP/IP, and send "*IDN?" command on NI-VISA to query the device information.
- Steps:

1. Open the MATLAB software, click "File>>New>>Script" on Matlab interface to create an empty M file.
2. Source code:

a) USBTMC Example

```
function usbtmc_test()
% This code demonstrates sending synchronous read & write commands
% to an USB Test & Measurement Class (USBTMC) instrument using
% NI-VISA

%Create a VISA-USB object connected to a USB instrument
vu = visa('ni','USB0::0x5345::0x1234::SN20220718::INSTR');

%Open the VISA object created
fopen(vu);

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

%Request the data
```

```
outputbuffer = fscanf(vu);
disp(outputbuffer);

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

end
```

b) TCP/IP Example

```
function tcp_ip_test()
% This code demonstrates sending synchronous read & write commands
% to an TCP/IP instrument using NI-VISA
%Create a VISA-TCPIP object connected to an instrument

%configured with IP address.
vt = visa('ni',['TCPIP0::','192.168.20.11','::inst0::INSTR']);

%Open the VISA object created

fopen(vt);

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

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

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

End
```

# Python Example

- Environment: Window System, Python3.8, and PyVISA 1.11.0.
- Description: Access the instrument via USBTMC and TCP/IP, and send “*IDN?” command on NI-VISA to query the device information.
- Steps:

1. Install python first, then open a Python script editor to create an empte test.py file.
2. Use the command “pip install PyVISA” to install PyVISA. If installation fails, refer to the following link (https://pyvisa.readthedocs.io/en/latest/).
3. Source code:

a) USBTMC Example

```
import pyvisa
rm = pyvisa.ResourceManager()
rm.list_resources()
my_instrument = rm.open_resource('USB0::0x5345::0x1234::SN20220718::INSTR')
print(my_instrument.query('*IDN?'))
```

b) TCP/TP Example

```
import pyvisa
rm = pyvisa.ResourceManager()
rm.list_resources()
my_instrument = rm.open_resource('TCPIP0::192.168.20.11::inst0::INSTR')
print(my_instrument.query('*IDN?')
```

# Programming Application Example

### (1) Set the bandwidth limits

When observing low-frequency signals, it's necessary to reduce the high-frequency noise in the signal, specifically by attenuating signals above 20 MHz.

Use the following command Set the bandwidth limits for Channel 1.

CHANnel1:BWLimit 20MHz # Enable the bandwidth limits 20 MHz for Channel 1.

CHANnel1:BWLimit? # The query returns 2.000000e+01.

### (2) Set bias voltage

Use the following command Set the bias voltage for Channel 1.

CHANnel1:OFFSet 1V # Channel 1 moves up by 1 V, setting the bias voltage to 1 V.

CHANnel1:OFFSet? # Query the bias voltage of Channel 1.

### (3) Set the volts/div scale

Use the following command Set the volts/div scale for Channel 1.

CHANnel1:SCALe 500mV # Set the volts/div scale of Channel 1 to 500 mV.

CHANnel1:SCALe? # Query the volts/div of Channel 1.

### (4) Set the time base scale

Use the following command Set the time base scale.

TIMebase:SCALe 0.005 # Set the time base scale of the oscilloscope to 5 ms.

TIMebase:SCALe? # Query the time base scale of the oscilloscope.

### (5) Query the amplitude

When querying the measured amplitude result, use the following command to query without opening the measurement window. For example, query the amplitude of Channel 1.

MEASure:ITEM? VPP,CHANnel1 # Query the amplitude of Channel 1 waveform.

### (6) Query the rising phase difference

When querying the phase difference between the rising edge of the master source and the rising edge of the slave source at the midpoint of the threshold, use the following command to query without opening the measurement window. For example, query the phase difference between the rising edges of Channel 1 and Channel 2.

MEASure:ITEM? RRPHase,CHANnel1,CHANnel2 # Query the rising edge phase difference between Channel 1 and Channel 2.