

# DPO2000 and MSO2000 Series Programmer Manual



077-0097-00

**Tektronix**



# DPO2000 and MSO2000 Series Programmer Manual

Copyright © Tektronix. All rights reserved. Licensed software products are owned by Tektronix or its subsidiaries or suppliers, and are protected by national copyright laws and international treaty provisions.

Tektronix products are covered by U.S. and foreign patents, issued and pending. Information in this publication supersedes that in all previously published material. Specifications and price change privileges reserved.

TEKTRONIX and TEK are registered trademarks of Tektronix, Inc.

### **Contacting Tektronix**

Tektronix, Inc.  
14200 SW Karl Braun Drive  
P.O. Box 500  
Beaverton, OR 97077  
USA

For product information, sales, service, and technical support:

- In North America, call 1-800-833-9200.
- Worldwide, visit [www.tektronix.com](http://www.tektronix.com) to find contacts in your area.

---

# Table of Contents

|  |      |
|--|------|
| Getting Started .....                          | 1-1  |
| Setting Up Remote Communications .....         | 1-1  |
| Command Syntax .....                           | 2-1  |
| Command and Query Structure .....              | 2-1  |
| Clearing the oscilloscope .....                | 2-4  |
| Command Entry .....                            | 2-4  |
| Constructed Mnemonics .....                    | 2-6  |
| Argument Types .....                           | 2-7  |
| Command Groups .....                           | 2-11 |
| Acquisition Command Group .....                | 2-11 |
| Alias Command Group .....                      | 2-12 |
| Bus Command Group .....                        | 2-13 |
| Calibration and Diagnostic Command Group ..... | 2-16 |
| Cursor Command Group .....                     | 2-17 |
| Display Command Group .....                    | 2-18 |
| Ethernet Command Group .....                   | 2-19 |
| File System Command Group .....                | 2-20 |
| FilterVu Command Group .....                   | 2-21 |
| Hard Copy Command Group .....                  | 2-22 |
| Horizontal Command Group .....                 | 2-23 |
| Mark Command Group .....                       | 2-23 |
| Math Command Group .....                       | 2-25 |
| Measurement Command Group .....                | 2-26 |
| Miscellaneous Command Group .....              | 2-29 |
| PictBridge Command Group .....                 | 2-30 |
| Save and Recall Command Group .....            | 2-31 |
| Search Command Group .....                     | 2-33 |
| Status and Error Command Group .....           | 2-37 |
| Trigger Command Group .....                    | 2-38 |
| Vertical Command Group .....                   | 2-45 |
| Waveform Transfer Command Group .....          | 2-48 |
| Zoom Command Group .....                       | 2-55 |
| Commands Listed in Alphabetical Order .....    | 2-57 |
| Status and Events .....                        | 3-1  |
| Registers .....                                | 3-1  |
| Queues .....                                   | 3-4  |
| Event Handling Sequence .....                  | 3-5  |
| Synchronization Methods .....                  | 3-7  |

|   |     |
|---|-----|
| Appendix A: Character Set .....                                   | A-1 |
| Appendix B: Waveform Data in DPO/MSO2000 Series Instruments ..... | B-1 |
| Appendix C: Reserved Words .....                                  | C-1 |
| Index   |     |

# Getting Started

This manual explains the use of commands for remotely controlling your oscilloscope. With this information, you can write computer programs to perform functions, such as setting the front-panel controls, taking measurements, performing statistical calculations, and exporting data for use in other programs.

## Setting Up Remote Communications

You can remotely communicate between your oscilloscope and PC via the Ethernet, USB, and, GPIB using the TEK-USB-488 Adapter.

### Ethernet

If you are using Ethernet, start by connecting an appropriate Ethernet cable to the Ethernet port (RJ-45 connector) on the rear panel of your oscilloscope. This connects the oscilloscope to a 10/100 Base-T local area network.



To change the Ethernet settings on your oscilloscope, do the following:

1. On the front panel, push **Utility**.
2. Push **Utility Page**.
3. Select **I/O** with the Multipurpose knob.
4. Push **Ethernet Network Settings**.
5. On the side-bezel menu, if you are on a DHCP Ethernet network and using a through cable, set DHCP/BOOTP to **On**.
6. If you are using a cross-over cable, set DHCP/BOOTP to **Off**, and set a hard coded TCP/IP address.

### USB

If you are using USB, start by connecting an appropriate USB cable to the USB 2.0 high-speed device port on the rear panel of your oscilloscope.



With USB, the system automatically configures itself. To verify that the USB is enabled:

1. On the front panel, push **Utility**.
2. Push **Utility Page**.
3. Select **I/O** with the Multipurpose knob.
4. Push **USB**, and verify that USB is enabled.
5. If USB is not enabled, push **Enabled** on the side-bezel menu.

After connection, the host will list the oscilloscope as a USB device with the following parameters. (See Table 1-1.)

**Table 1-1: USB Device Parameters**

| Parameter                | Value  |
|--------------------------|--|
| Manufacturer ID          | 0x0699 (decimal 1689)  |
| Product ID               | 0x0372 DPO2012<br>0x0373 DPO2014<br>0x0374 DPO2024<br>0x0376 MSO2012<br>0x0377 MSO2014<br>0x0378 MSO2024 |
| Serial number            | Serial number  |
| Manufacturer description | "Tektronix"  |
| Interface description    | "USBTMC-USB488"  |

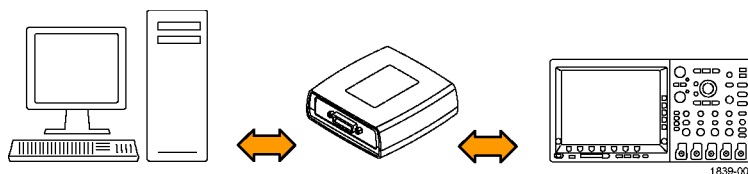
**GPIB** To use GPIB, start by connecting an appropriate USB cable to the USB 2.0 high-speed device port on the rear panel of your oscilloscope. Connect the other end to the TEK-USB-488 Adapter host port. Then connect a GPIB cable from the TEK-USB-488 Adapter to your PC.



Supply power to the Adapter in either of these two ways:

1. Use the optional 5 V<sub>DC</sub> power adapter connected to the 5 V<sub>DC</sub> power input on the Adapter.
2. Use an appropriate USB cable connected to a powered USB host port on your PC and the Device port on the TEK-USB-488 Adapter.

The oscilloscope has a USB 2.0 high-speed device port to control the oscilloscope through USBTMC or GPIB with a TEK-USB-488 Adapter. The USBTMC protocol allows USB devices to communicate using IEEE488 style messages. This lets you run your GPIB software applications on USB hardware.



Before setting up the oscilloscope for remote communication using the electronic (physical) GPIB interface, you should familiarize yourself with the following GPIB requirements:

- A unique device address must be assigned to each device on the bus. No two devices can share the same device address.
- No more than 15 devices can be connected to any one line.
- Only one device should be connected for every 6 feet (2 meters) of cable used.
- No more than 65 feet (20 meters) of cable should be used to connect devices to a bus.
- At least two-thirds of the devices on the network should be powered on while using the network.
- Connect the devices on the network in a star or linear configuration. Do not use loop or parallel configurations.

To function correctly, your oscilloscope must have a unique device address. The default setting for the GPIB configuration is GPIB Address 1.

To change the GPIB address settings, do the following:

1. On the front panel, push **Utility**.
2. Push **Utility Page**.
3. Select **I/O** with the Multipurpose knob.
4. Push **GPIB**.
5. Enter the GPIB address on the side-bezel menu, using the multipurpose knob. This will set the GPIB address on an attached TEK-USB-488 Adapter

The oscilloscope is now set up for bidirectional communication with your controller.

### Documentation

The following documents are available for download on the Manuals Finder Web site at [www.tektronix.com](http://www.tektronix.com):

***DPO/MSO2000 Series User Manual.*** Information about installing and operating the oscilloscope.

***Getting Started with OpenChoice™ Solutions Manual.*** Options for getting data from your oscilloscope into any one of several available analysis tools.

***DPO/MSO2000 Series Technical Reference.*** Oscilloscope specifications and a performance verification procedure.

***TekVISA Programmer Manual.*** Description of TekVISA, the Tektronix implementation of the VISA Application Programming Interface (API). TekVISA is industry-compliant software for writing interoperable oscilloscope drivers in a variety of Application Development Environments (ADEs).

# Command Syntax

You can control the operations and functions of the oscilloscope through the Ethernet port or the USB 2.0 device port using commands and queries. The related topics listed below describe the syntax of these commands and queries. The topics also describe the conventions that the oscilloscope uses to process them. See the *Command Groups* topic in the table of contents for a listing of the commands by command group, or use the index to locate a specific command.

## Backus-Naur Form Notation

This documentation describes the commands and queries using Backus-Naur Form (BNF) notation. Refer to the following table for the symbols that are used.

**Table 2-1: Symbols for Backus-Naur Form**

| Symbol | Meaning                             |
|--------|-------------------------------------|
| < >    | Defined element                     |
| =      | Is defined as                       |
|        | Exclusive OR                        |
| { }    | Group; one element is required      |
| [ ]    | Optional; can be omitted            |
| . . .  | Previous element(s) may be repeated |
| ( )    | Comment                             |

## Command and Query Structure

Commands consist of set commands and query commands (usually called commands and queries). Commands modify oscilloscope settings or tell the oscilloscope to perform a specific action. Queries cause the oscilloscope to return data and status information.

Most commands have both a set form and a query form. The query form of the command differs from the set form by its question mark at the end. For example, the set command `ACQUIRE:MODE` has a query form `ACQUIRE:MODE?`. Not all commands have both a set and a query form. Some commands have set only and some have query only.

## Messages

A command message is a command or query name followed by any information the oscilloscope needs to execute the command or query. Command messages may contain five element types, defined in the following table.

**Table 2-2: Command Message Elements**

| <b>Symbol</b> | <b>Meaning</b>   |
|---------------|--|
| <Header>      | This is the basic command name. If the header ends with a question mark, the command is a query. The header may begin with a colon (:) character. If the command is concatenated with other commands, the beginning colon is required. Never use the beginning colon with command headers beginning with a star (*). |
| <Mnemonic>    | This is a header subfunction. Some command headers have only one mnemonic. If a command header has multiple mnemonics, a colon (:) character always separates them from each other.  |
| <Argument>    | This is a quantity, quality, restriction, or limit associated with the header. Some commands have no arguments while others have multiple arguments. A <space> separates arguments from the header. A <comma> separates arguments from each other.   |
| <Comma>       | A single comma is used between arguments of multiple-argument commands. Optionally, there may be white space characters before and after the comma.  |
| <Space>       | A white space character is used between a command header and the related argument. Optionally, a white space may consist of multiple white space characters.   |

**Commands**

Commands cause the oscilloscope to perform a specific function or change one of the settings. Commands have the structure:

[ : ] <Header> [ <Space> <Argument> [ <Comma> <Argument> ] . . . ]

A command header consists of one or more mnemonics arranged in a hierarchical or tree structure. The first mnemonic is the base or root of the tree and each subsequent mnemonic is a level or branch off the previous one. Commands at a higher level in the tree may affect those at a lower level. The leading colon (:) always returns you to the base of the command tree.

**Queries** Queries cause the oscilloscope to return status or setting information. Queries have the structure:

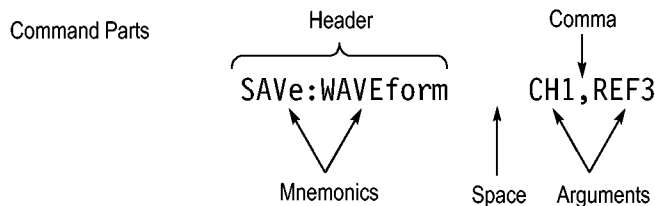
- [ : ] <Header>
- [ : ] <Header> [ <Space> <Argument> [ <Comma> <Argument> ] . . . ]

You can specify a query command at any level within the command tree unless otherwise noted. These branch queries return information about all the mnemonics below the specified branch or level.

**Headers** Use the HEADer command to control whether the oscilloscope returns headers as part of the query response. If header is on, the query response returns command headers, then formats itself as a valid set command. When header is off, the response includes only the values. This may make it easier to parse and extract the information from the response. The table below shows the difference in responses.

**Table 2-3: Comparison of Header Off and Header On Responses**

| Query           | Header Off | Header On           |
|-----------------|------------|---------------------|
| TIME?           | 14:30:00   | :TIME "14:30:00"    |
| ACQuire:NUMAVg? | 100        | :ACQUIRE:NUMAVG 100 |



## Clearing the oscilloscope

You can clear the Output Queue and reset the oscilloscope to accept a new command or query by using the selected Device Clear (DCL) function.

## Command Entry

The following rules apply when entering commands:

- You can enter commands in upper or lower case.
- You can precede any command with white space characters. White space characters include any combination of the ASCII control characters 00 through 09 and 0B through 20 hexadecimal (0 through 9 and 11 through 32 decimal).
- The oscilloscope ignores commands consisting of any combination of white space characters and line feeds.

### Abbreviating

You can abbreviate many oscilloscope commands. Each command in this documentation shows the minimum acceptable abbreviations in capitals. For example, you can enter the command ACQuire:NUMAvg simply as ACQ:NUMA or acq:numa.

Abbreviation rules may change over time as new oscilloscope models are introduced. Thus, for the most robust code, use the full spelling.

If you use the HEADer command to have command headers included as part of query responses, you can further control whether the returned headers are abbreviated or are full-length with the VERBose command.

### Concatenating

You can concatenate any combination of set commands and queries using a semicolon (;). The oscilloscope executes concatenated commands in the order received.

When concatenating commands and queries, you must follow these rules:

1. Separate completely different headers by a semicolon and by the beginning colon on all commands except the first one. For example, the commands `TRIGger:MODE NORMAl` and `ACQuire:NUMAVg 8`, can be concatenated into the following single command:

```
TRIGger:MODE NORMAl;:ACQuire:NUMAVg 8
```

2. If concatenated commands have headers that differ by only the last mnemonic, you can abbreviate the second command and eliminate the beginning colon. For example, you can concatenate the commands `ACQuire:MODE AVErAge` and `ACQuire:NUMAVg 8` into a single command:

```
ACQuire:MODE AVErAge; NUMAVg 8
```

The longer version works equally well:

```
ACQuire:MODE AVErAge;:ACQuire:NUMAVg 8
```

3. Never precede a star (\*) command with a colon:

```
ACQuire:STATE 1;*OPC
```

Any commands that follow will be processed as if the star command was not there so the commands, `ACQuire:MODE ENVElope;*OPC;NUMAVg 8` will set the acquisition mode to envelope and set the number of acquisitions for averaging to 8.

4. When you concatenate queries, the responses to all the queries are concatenated into a single response message.
5. Set commands and queries may be concatenated in the same message. For example,

```
ACQuire:MODE SAMple;NUMAVg?;STATE?
```

is a valid message that sets the acquisition mode to sample. The message then queries the number of acquisitions for averaging and the acquisition state. Concatenated commands and queries are executed in the order received.

Here are some invalid concatenations:

```
HORizontal:SCALE 400E-9;ACQuire:NUMAVg 8 (no colon before ACQuire)
```

```
DISPlay:GRAticule FULL;:*TRG (colon before a star (*) command)
```

```
MATH:HORizontal:SCALE 1.0e-1;HORizontal:POSITION 5.0e1  
(levels of the mnemonics are different; either remove the second use of HORizontal: or place :MATH in front of HORizontal:POSITION)
```

### Terminating

This documentation uses <EOM> (End of Message) to represent a message terminator.

**Table 2-4: End of Message Terminator**

| Symbol | Meaning            |
|--------|--------------------|
| <EOM>  | Message terminator |

The end-of-message terminator must be the END message (EOI asserted concurrently with the last data byte). The last data byte may be an ASCII line feed (LF) character.

This oscilloscope does not support ASCII LF only message termination. The oscilloscope always terminates outgoing messages with LF and EOI.

## Constructed Mnemonics

Some header mnemonics specify one of a range of mnemonics. For example, a channel mnemonic can be CH1, CH2, CH3, or CH4. You use these mnemonics in the command just as you do any other mnemonic. For example, there is a CH1:POSition command, and there is also a CH2:POSition command. In the command descriptions, this list of choices is abbreviated as CH<x>.

### Cursor Position Mnemonics

When cursors are displayed, commands may specify which cursor of the pair to use.

**Table 2-5: Channel Mnemonics**

| Symbol | Meaning                                  |
|--------|--|
| CH<x>  | A channel specifier; <x> is 1 through 4. |

**Table 2-6: Cursor Mnemonics**

| Symbol      | Meaning                                  |
|-------------|--|
| CURSOR<x>   | A cursor selector; <x> is either 1 or 2. |
| POSITION<x> | A cursor selector; <x> is either 1 or 2. |
| HPOS<x>     | A cursor selector; <x> is either 1 or 2. |

### Math Specifier Mnemonics

Commands can specify the mathematical waveform to use as a mnemonic in the header.

**Table 2-7: Math Specifier Mnemonics**

| Symbol  | Meaning                              |
|---------|--------------------------------------|
| Math<x> | A math waveform specifier; <x> is 1. |



**Measurement Specifier Mnemonics** Commands can specify which measurement to set or query as a mnemonic in the header. Up to four automated measurements may be displayed.

**Table 2-8: Measurement Specifier Mnemonics**

| Symbol  | Meaning                                      |
|---------|--|
| MEAS<x> | A measurement specifier; <x> is 1 through 4. |

**Channel Mnemonics** Commands specify the channel to use as a mnemonic in the header.

**Reference Waveform Mnemonics** Commands can specify the reference waveform to use as a mnemonic in the header.

**Table 2-9: Reference Waveform Mnemonics**

| Symbol | Meaning  |
|--------|--|
| REF<x> | A reference waveform specifier; <x> 1 or 2. The DPO/MSO2000 series provides only two REF waveforms regardless of whether the instrument is a 2 or 4 channel model. |

## Argument Types

**Numeric** Many oscilloscope commands require numeric arguments. The syntax shows the format that the oscilloscope returns in response to a query. This is also the preferred format when sending the command to the oscilloscope though any of the formats will be accepted. This documentation represents these arguments as described below.

**Table 2-10: Numeric Arguments**

| Symbol | Meaning                                  |
|--------|--|
| <NR1>  | Signed integer value                     |
| <NR2>  | Floating point value without an exponent |
| <NR3>  | Floating point value with an exponent    |
| <bin>  | Digital data in binary format            |

Most numeric arguments will be automatically forced to a valid setting, by either rounding or truncating, when an invalid number is input, unless otherwise noted in the command description.

**Quoted String** Some commands accept or return data in the form of a quoted string, which is simply a group of ASCII characters enclosed by a single quote (') or double quote ("). The following is an example of a quoted string: "This is a quoted string". This documentation represents these arguments as follows:

**Table 2-11: Quoted String Argument**

| Symbol    | Meaning                     |
|-----------|-----------------------------|
| <QString> | Quoted string of ASCII text |

A quoted string can include any character defined in the 7-bit ASCII character set. Follow these rules when you use quoted strings:

1. Use the same type of quote character to open and close the string. For example: "this is a valid string".
2. You can mix quotation marks within a string as long as you follow the previous rule. For example: "this is an 'acceptable' string".
3. You can include a quote character within a string by repeating the quote. For example: "here is a "" mark".
4. Strings can have upper or lower case characters.
5. If you use a GPIB network, you cannot terminate a quoted string with the END message before the closing delimiter.
6. A carriage return or line feed embedded in a quoted string does not terminate the string. The return is treated as another character in the string.
7. The maximum length of a quoted string returned from a query is 1000 characters.

Here are some invalid strings:

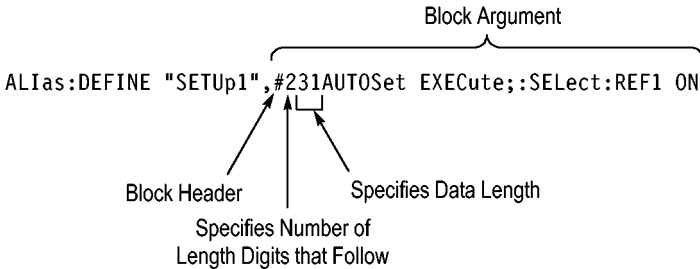
- "Invalid string argument' (quotes are not of the same type)
- "test<EOI>" (termination character is embedded in the string)

**Block** Several oscilloscope commands use a block argument form, as defined in the table below.

**Table 2-12: Block Argument**

| Symbol  | Meaning   |
|---------|---|
| <NZDig> | A nonzero digit character in the range of 1–9   |
| <Dig>   | A digit character, in the range of 0–9  |
| <DChar> | A character with the hexadecimal equivalent of 00 through FF (0 through 255 decimal)                            |
| <Block> | A block of data bytes defined as: <Block> ::= {#<NZDig><Dig>[<Dig>...][<DChar>...]  #0[<DChar>...]<terminator>} |

<NZDig> specifies the number of <Dig> elements that follow. Taken together, the <NZDig> and <Dig> elements form a decimal integer that specifies how many <DChar> elements follow.





---

# Command Groups

This manual lists the DPO/MSO2000 series IEEE488.2 commands in two ways. First, it presents them by functional groups. Then, it lists them alphabetically. The functional group list starts below. The alphabetical list provides detail on each command. (See page 2-57, *Commands Listed in Alphabetical Order.*)

## Acquisition Command Group

Use the commands in the Acquisition Command Group to set up the modes and functions that control how the oscilloscope acquires signals input to the channels, and processes them into waveforms.

Using the commands in this group, you can do the following:

- Start and stop acquisitions.
- Control whether each waveform is simply acquired, averaged over successive acquisitions of that waveform.
- Set the controls or conditions that start and stop acquisitions.
- Control acquisition of channel waveforms.
- Set acquisition parameters.

**Table 2-13: Acquisition Commands**

| Command                                | Description  |
|--|--|
| <a href="#">ACQUIRE?</a>               | Returns acquisition parameters   |
| <a href="#">ACQUIRE:MAXSAMPLERATE?</a> | Returns the maximum real-time sample rate                                |
| <a href="#">ACQUIRE:MODE</a>           | Sets or returns the acquisition mode                                     |
| <a href="#">ACQUIRE:NUMACQ?</a>        | Returns number of acquisitions that have occurred                        |
| <a href="#">ACQUIRE:NUMAVG</a>         | Sets or returns the number of acquisitions for an averaged waveform      |
| <a href="#">ACQUIRE:STATE</a>          | Starts or stops the acquisition system                                   |
| <a href="#">ACQUIRE:STOPAFTER</a>      | Sets or returns whether the acquisition is continuous or single sequence |

## Alias Command Group

Use the Alias commands to define new commands as a sequence of standard commands. You may find this useful when repeatedly using the same commands to perform certain tasks like setting up measurements.

Aliases are similar to macros but do not include the capability to substitute parameters into alias bodies. The alias mechanism obeys the following rules:

- The alias name must consist of a valid IEEE488.2 message unit, which may not appear in a message preceded by a colon, comma, or a command or query program header.
- The alias name may not appear in a message followed by a colon, comma, or question mark.
- An alias name must be distinct from any keyword or keyword short form.
- An alias name cannot be redefined without first being deleted using one of the alias deletion functions.
- Alias names do not appear in response messages.

**Table 2-14: Alias Commands**

| <b>Command</b>                      | <b>Description</b>                                       |
|-------------------------------------|--|
| <a href="#">ALias</a>               | Sets or returns the alias state                          |
| <a href="#">ALias:CATalog?</a>      | Returns a list of the currently defined alias labels     |
| <a href="#">ALias:DEFine</a>        | Assigns a sequence of program messages to an alias label |
| <a href="#">ALias:DELEte</a>        | Removes a specified alias                                |
| <a href="#">ALias:DELEte:ALL</a>    | Deletes all existing aliases                             |
| <a href="#">ALias:DELEte[:NAME]</a> | Removes a specified alias                                |
| <a href="#">ALias[:STATE]</a>       | Sets or returns the alias state                          |

## Bus Command Group

Use the Bus commands when working with serial bus measurements.

- Install the DPO2EMBD application module when working with I<sup>2</sup>C or SPI bus signals.
- Install the DPO2AUTO module when working with CAN or LIN bus signals.
- Install the DPO2COMP module when working with RS232 bus signals.

**Table 2-15: Bus Commands**

| Commands  | Description   |
|---|---|
| <a href="#">BUS</a>   | Returns the parameters for each bus   |
| <a href="#">BUS:B&lt;x&gt;:CAN:BITRate</a>                  | Sets or returns the bit rate for the CAN bus  |
| <a href="#">BUS:B&lt;x&gt;:CAN:PRObe</a>                    | Sets or returns the probing method used to probe the CAN bus                        |
| <a href="#">BUS:B&lt;x&gt;:CAN:SAMPLEpoint</a>              | Sets or returns the sample point (in %) to sample during each bit period            |
| <a href="#">BUS:B&lt;x&gt;:CAN:SOURce</a>                   | Sets or returns the CAN data source   |
| <a href="#">BUS:B&lt;x&gt;:DISplay:FORMAt</a>               | Sets the display format for the numerical information in the specified bus waveform |
| <a href="#">BUS:B&lt;x&gt;:I2C:ADDRess:RWINClude</a>        | Sets and returns whether the read/write bit is included in the address              |
| <a href="#">BUS:B&lt;x&gt;:I2C{:CLOCK SCLK}:SOURce</a>      | Sets or returns the I2C SCLK source   |
| <a href="#">BUS:B&lt;x&gt;:I2C{:DATA SDATA}:SOURce</a>      | Sets or returns the I2C SDATA source  |
| <a href="#">BUS:B&lt;x&gt;:LABel</a>                        | Sets or returns the waveform label for the specified bus                            |
| <a href="#">BUS:B&lt;x&gt;:LIN:BITRate</a>                  | Sets or returns the bit rate for LIN  |
| <a href="#">BUS:B&lt;x&gt;:LIN:IDFORmat</a>                 | Sets or returns the LIN ID format   |
| <a href="#">BUS:B&lt;x&gt;:LIN:POLARity</a>                 | Sets or returns the LIN polarity  |
| <a href="#">BUS:B&lt;x&gt;:LIN:SAMPLEpoint</a>              | Sets or returns the sample point (in %) at which to sample during each bit period   |
| <a href="#">BUS:B&lt;x&gt;:LIN:SOURce</a>                   | Sets or returns the LIN data source   |
| <a href="#">BUS:B&lt;x&gt;:LIN:STANDard</a>                 | Sets or returns the LIN standard  |
| <a href="#">BUS:B&lt;x&gt;:PARAllel:BIT&lt;x&gt;:SOURce</a> | Sets or returns the parallel bit <x> source   |
| <a href="#">BUS:B&lt;x&gt;:PARAllel:CLOCK:EDGE</a>          | Sets or returns the parallel clock edge for bus <x>                                 |
| <a href="#">BUS:B&lt;x&gt;:PARAllel:CLOCK:ISCLOCKed</a>     | Sets or returns whether the parallel bus is clocked                                 |
| <a href="#">BUS:B&lt;x&gt;:PARAllel:CLOCK:SOURce</a>        | Sets or returns the parallel bus<x> clock source                                    |
| <a href="#">BUS:B&lt;x&gt;:PARAllel:WIDth</a>               | Sets or returns the number of bits used for the width of the parallel bus <x>       |

Table 2-15: Bus Commands (cont.)

| Commands                               | Description  |
|--|--|
| BUS:B<x>:POsition                      | Sets or returns the position of the specified bus waveform                     |
| BUS:B<x>:RS232C:BITRate                | Sets or returns the RS232C bit rate for the specified bus                      |
| BUS:B<x>:RS232C:DATABits               | Sets or returns the number of bits for the data frame                          |
| BUS:B<x>:RS232C:DELIMiter              | Sets or returns the RS232C delimiting value for a packet on the specified bus  |
| BUS:B<x>:RS232C:DISplaymode            | Sets or returns the display mode for the specified bus display and event table |
| BUS:B<x>:RS232C:PARity                 | Sets or returns parity for RS232C data   |
| BUS:B<x>:RS232C:POLarity               | Sets or returns the RS232C polarity for the specified bus                      |
| BUS:B<x>:RS232C:RX:SOURce              | Sets or returns the RS232C RX source   |
| BUS:B<x>:RS232C:TX:SOURce              | Sets or returns the RS232C TX Source   |
| BUS:B<x>:SPI{;CLOCK ;SCLK}:POLARity    | Sets or returns the SPI SCLK polarity  |
| BUS:B<x>:SPI{;CLOCK ;SCLK}:SOURce      | Sets or returns the SPI SCLK source  |
| BUS:B<x>:SPI:DATA{;IN ;MISO}:POLARity  | Sets or returns the SPI MISO polarity  |
| BUS:B<x>:SPI:DATA{;IN ;MISO}:SOURce    | Sets or returns the SPI MISO source  |
| BUS:B<x>:SPI:DATA{;OUT ;MOSI}:POLARity | Sets or returns the SPI MOSI polarity  |
| BUS:B<x>:SPI:DATA{;OUT ;MOSI}:SOURce   | Sets or returns the SPI MOSI source  |
| BUS:B<x>:SPI{;SElect ;SS}:POLARity     | Sets or returns the SPI SS polarity  |
| BUS:B<x>:SPI{;SElect ;SS}:SOURce       | Sets or returns the SPI SS source  |
| BUS:B<x>:SPI:BITOrder                  | Sets or returns the bit order for the specified SPI bus                        |
| BUS:B<x>:SPI:DATA:SIZE                 | Sets or returns the number of bits per word for the specified SPI bus          |
| BUS:B<x>:SPI:FRAMING                   | Sets or returns the type of SPI framing  |
| BUS:B<x>:SPI:IDLETime                  | Sets or returns the SPI bus idle time in seconds for the specified SPI bus     |
| BUS:B<x>:STATE                         | Turns the specified bus on and off   |
| BUS:B<x>:TYPE                          | Sets or returns the specified bus type   |
| BUS:LOWerthreshold:CH<x>               | Sets or returns the lower threshold for each channel                           |
| BUS:THReshold:CH<x>                    | Sets or returns the threshold for a channel                                    |
| BUS:UPPerthreshold:CH<x>               | Sets or returns the upper threshold for each channel                           |
| BUS:THReshold:D<x>                     | Sets or returns the threshold for digital channel                              |



Table 2-15: Bus Commands (cont.)

| Commands   | Description   |
|--|---|
| SEARCH:SEARCH<x>:TRIGger:A:BUS?                              | Returns the serial search type  |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:LIN:CONDition        | Sets or returns the search condition for a LIN search                             |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:LIN:DATA:HIVALue     | Sets or returns the binary data string  |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:LIN:DATA:QUALifier   | Sets or returns the LIN data qualifier  |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:LIN:DATA:SIZE        | Sets or returns the length of the data string in bytes                            |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:LIN:DATA:VALue       | Sets or returns the binary data string used for a LIN search                      |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:LIN:ERRTYPE          | Sets or returns the error type used for a LIN Search                              |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:LIN:IDentifier:VALue | Sets or returns the binary address string used for LIN search                     |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:RS232C:CONDition     | Sets or returns the trigger condition for a RS232 trigger                         |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:RS232C:RX:DATA:SIZE  | Sets or returns the length of the data string for a RS232 RX trigger              |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:<br>B<x>:RS232C:RX:DATA:VALue | Sets or returns the binary data string for a RX RS232 trigger                     |
| TRIGger:A:BUS:B<x>:LIN:CONDition                             | Sets or returns the trigger condition for LIN                                     |
| TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue                          | Sets or returns the binary data string to be used for LIN trigger                 |
| TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier                        | Sets or returns the LIN data qualifier  |
| TRIGger:A:BUS:B<x>:LIN:DATA:SIZE                             | Sets or returns the length of the data string in bytes to be used for LIN trigger |
| TRIGger:A:BUS:B<x>:LIN:DATA:VALue                            | Sets or returns the binary data string  |
| TRIGger:A:BUS:B<x>:LIN:ERRTYPE                               | Sets or returns the error type  |
| TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue                      | Sets or returns the binary address string used for LIN trigger                    |
| TRIGger:A:BUS:B<x>:RS232C:RX:DATA:<br>SIZE                   | Sets or returns the length of the data string for a RX RS232 trigger              |
| TRIGger:A:BUS:B<x>:RS232C:RX:DATA:<br>VALue                  | Sets or returns the binary data string for a RX RS232 trigger                     |
| TRIGger:A:BUS:B<x>:RS232C:TX:DATA:<br>SIZE                   | Sets or returns the length of the data string to be used for a TX RS232 Trigger   |
| TRIGger:A:BUS:B<x>:RS232C:TX:DATA:<br>VALue                  | Sets or returns the binary data string to be used for a TX RS232 trigger          |

## Calibration and Diagnostic Command Group

The Calibration and Diagnostic commands provide information about the current state of oscilloscope calibration. They also initiate internal signal path calibration (SPC) or execute diagnostic tests. Commands that are specific to factory calibration are not described in this manual. They are described in the Service manual, located on the DPO2000 Documentation CD-ROM in PDF format. You can also order a printed copy.

**Table 2-16: Calibration and Diagnostic Commands**

| Command                     | Description   |
|-----------------------------|---|
| *CAL?                       | Instructs the oscilloscope to perform self-calibration and returns the oscilloscope self calibration status |
| CALibrate:FACTory           | Provides the controls for starting and stopping the factory calibration process                             |
| CALibrate:FACTory:STATus?   | Returns the factory calibration status value saved in nonvolatile memory                                    |
| CALibrate:INTERNAal         | Starts a signal path compensation   |
| CALibrate:INTERNAal:START   | Starts the internal signal path calibration   |
| CALibrate:INTERNAal:STATus? | Returns the current status of the internal signal path calibration  |
| CALibrate:RESults?          | Returns the status of all calibration subsystems without performing an SPC operation                        |
| CALibrate:RESults:FACTory?  | Returns the status of internal and factory calibration  |
| CALibrate:RESults:SPC?      | Returns the results of the last SPC operation   |
| CALibrate:TEMPerature?      | Returns 0 as DPO/MSO2000 series does not support recording of oscilloscope temperature                      |
| DIAg:LOOP:OPTion            | Sets the self-test loop option  |
| DIAg:LOOP:OPTion:NTIMes     | Sets the self-test loop option to run N times   |
| DIAg:LOOP:STOP              | Stops the self-test at the end of the current loop  |
| DIAg:RESUlt:FLAg?           | Returns the pass/fail status from the last self-test sequence execution                                     |
| DIAg:RESUlt:LOG?            | Returns the internal results log from the last self-test sequence execution                                 |
| DIAg:SELEct:<function>      | Selects one of the available self-test areas  |
| DIAg:STATE                  | Sets the oscilloscope operating state   |
| DIAg:SELEct                 | Runs self tests on the specified system subsystem   |

## Cursor Command Group

Use the commands in the Cursor Command Group to control the cursor display and readout. You can use these commands to control the setups for cursor 1 and cursor 2, such as cursor position.

You can also use the commands to select one of the following cursor functions:

- Off. Turns off the display of all cursors.
- Waveform Cursors. Consists of two cursors. Waveform cursors enable you to conveniently measure waveform amplitude and time.
- Screen Cursors. Consists of two pairs of independent horizontal and vertical cursors. You can use these cursors to indicate an arbitrary position within the waveform display area.

**Table 2-17: Cursor Commands**

| Command  | Description   |
|--|---|
| <a href="#">CURSor?</a>                          | Returns cursor settings   |
| <a href="#">CURSor:FUNCTion</a>                  | Sets or returns the cursor type   |
| <a href="#">CURSor:HBArs?</a>                    | Returns hbar cursor settings  |
| <a href="#">CURSor:HBArs:DELta?</a>              | Returns hbars cursors vertical difference                                   |
| <a href="#">CURSor:HBArs:POSITION&lt;x&gt;</a>   | Sets or returns the hbar cursor<x> vertical position                        |
| <a href="#">CURSor:HBArs:UNIts</a>               | Returns hbar cursor units   |
| <a href="#">CURSor:HBArs:USE</a>                 | Sets the horizontal bar cursor measurement scale                            |
| <a href="#">CURSor:MODe</a>                      | Sets or returns whether cursors move in unison or separately                |
| <a href="#">CURSor:VBArS?</a>                    | Sets or returns the position of vertical bar cursors                        |
| <a href="#">CURSor:VBArS:ALTERNATE&lt;x&gt;?</a> | Returns the alternate readout for the waveform (Vbar) cursors               |
| <a href="#">CURSor:VBArS:DELta?</a>              | Returns the difference between vbar cursors                                 |
| <a href="#">CURSor:VBArS:HPOS&lt;x&gt;?</a>      | Returns the horizontal value of the specified vertical bar ticks            |
| <a href="#">CURSor:VBArS:POSITION&lt;x&gt;</a>   | Sets or returns the vbar cursor<x> horizontal position                      |
| <a href="#">CURSor:VBArS:UNIts</a>               | Sets or returns the units for vbar cursors                                  |
| <a href="#">CURSor:VBArS:USE</a>                 | Sets the vertical bar cursor measurement scale                              |
| <a href="#">CURSor:VBArS:VDELta?</a>             | Returns the vertical difference between the two vertical bar cursor ticks   |
| <a href="#">CURSor:XY:POLar:RADIUS:DELta?</a>    | Returns the difference between the cursors X radius and the cursor Y radius |

Table 2-17: Cursor Commands (cont.)

| Command   | Description  |
|---|--|
| <a href="#">CURSor:XY:POLar:RADIUS:POSITION&lt;x&gt;?</a> | Returns the polar radius of the specified cursor   |
| <a href="#">CURSor:XY:POLar:RADIUS:UNIts?</a>             | Returns the polar radius units   |
| <a href="#">CURSor:XY:POLar:THETA:DELta?</a>              | Returns the XY cursor polar coordinate delta   |
| <a href="#">CURSor:XY:POLar:THETA:POSITION&lt;x&gt;?</a>  | Returns the cursor X or cursor Y polar coordinate  |
| <a href="#">CURSor:XY:POLar:THETA:UNIts?</a>              | Returns the cursor polar coordinate units  |
| <a href="#">CURSor:XY:PRODUCT:DELta?</a>                  | Returns the difference between the cursors X position and cursor Y position              |
| <a href="#">CURSor:XY:PRODUCT:POSITION&lt;x&gt;?</a>      | Returns the position of the X or Y cursor used to calculate the X × Y cursor measurement |
| <a href="#">CURSor:XY:PRODUCT:UNIts?</a>                  | Returns the XY cursor product units  |
| <a href="#">CURSor:XY:RATIO:DELta?</a>                    | Returns the ratio of the difference between the cursor X position and cursor Y position  |
| <a href="#">CURSor:XY:RATIO:POSITION&lt;x&gt;?</a>        | Returns the X or Y position for the specified cursor                                     |
| <a href="#">CURSor:XY:RATIO:UNIts?</a>                    | Returns the X and Y cursor units for the ratio measurement                               |
| <a href="#">CURSor:XY:RECTangular:X:DELta?</a>            | Returns the cursor X delta value in rectangular coordinates                              |
| <a href="#">CURSor:XY:RECTangular:X:POSITION&lt;x&gt;</a> | Sets or returns the cursor X rectangular coordinates                                     |
| <a href="#">CURSor:XY:RECTangular:X:UNIts?</a>            | Returns the Cursor X rectangular units   |
| <a href="#">CURSor:XY:RECTangular:Y:DELta?</a>            | Returns The cursor Y delta value in rectangular coordinates                              |
| <a href="#">CURSor:XY:RECTangular:Y:POSITION&lt;x&gt;</a> | Sets or returns the cursor Y rectangular coordinates                                     |
| <a href="#">CURSor:XY:RECTangular:Y:UNIts?</a>            | Returns the cursor Y rectangular units   |

## Display Command Group

Use the commands in the Display Command Group to change the graticule style, the displayed intensities, and to set the characteristics of the waveform display.

Use these commands to set the style that best displays your waveforms and graticule display properties. Note that the mode you choose globally affects all displayed waveforms.

Table 2-18: Display Commands

| Command                  | Description                      |
|--------------------------|----------------------------------|
| <a href="#">DISplay?</a> | Returns current display settings |

Table 2-18: Display Commands (cont.)

| Command                     | Description  |
|-----------------------------|--|
| DISplay:CLOCK               | Sets or returns the display of the date/time stamp   |
| DISplay:DIGital:HEIght      | Sets or returns the height of the digital display and the number of waveforms that you can display |
| DISplay:FORMat              | Sets or returns the display format   |
| DISplay:GRAticule           | Sets or returns the type of graticule that is displayed  |
| DISplay:INTENSITy?          | Returns all display intensity settings   |
| DISplay:INTENSITy:BACKLight | Sets or returns the backlight intensity for the display  |
| DISplay:INTENSITy:GRAticule | Sets or returns the graticule intensity for the display  |
| DISplay:INTENSITy:WAVEform  | Sets or returns the intensity of the waveforms   |
| DISplay:PERsistence         | Sets or returns display persistence setting  |
| MESSage:BOX                 | Sets or returns the size and position of the message window  |
| MESSage:CLEAR               | Removes the message text from the message window   |
| MESSage:SHOW                | Clears the contents of the message window  |
| MESSage:STATE               | Controls the display of the message window   |

## Ethernet Command Group

Use the commands in the Ethernet Command Group to set up the Ethernet remote interface.

Table 2-19: Ethernet Commands

| Command                    | Description   |
|----------------------------|---|
| ETHERnet:DHCPbootp         | Sets or returns the network initialization search for a DHCP/BOOTP server |
| ETHERnet:DNS:IPADDRESS     | Sets or returns the network Domain Name Server (Dns) IP address           |
| ETHERnet:DOMAINname        | Sets or returns the network domain name                                   |
| ETHERnet:ENET:ADDRESS?     | Returns the Ethernet address value assigned to the oscilloscope           |
| ETHERnet:GATEWay:IPADDRESS | Sets or returns the remote interface gateway IP address                   |
| ETHERnet:HTTTPort          | Sets or returns the remote interface HTTP port value                      |

Table 2-19: Ethernet Commands (cont.)

| Command               | Description   |
|-----------------------|---|
| ETHERnet:IPADdress    | Sets or returns the IP address assigned to the oscilloscope   |
| ETHERnet:NAME         | Sets or returns the network name assigned to the oscilloscope |
| ETHERnet:PASSWord     | Sets or returns the Ethernet access password                  |
| ETHERnet:PING         | Causes the oscilloscope to ping the gateway IP address        |
| ETHERnet:PING:STATUS? | Returns the results from pinging the gateway IP address       |
| ETHERnet:SUBNETMask   | Sets or returns the remote interface subnet mask value        |

## File System Command Group

Use the commands in the File System Command Group to access USB media. You can use the commands to do the following:

- List the contents of a directory
- Create, rename and delete directories
- Create, read, rename, or delete a file
- Format media

When using these commands, keep the following points in mind:

- File arguments are always enclosed within double quotes:  
"E:/MYDIR/TEK00001.SET"
- File names follow the non-case sensitive, MSDOS format:  
[DRIVE:][\PATH\]filename
- File names for commands and queries are not case sensitive. Save commands translate all file names to uppercase for storage

For Example: SAVE:WAVEFORM CH1,"ch1.isf" results in Ch1 being saved to a file named CH1.ISF.

RECALL:WAVEFORM "Ch1.isf",REF1 would recall the waveform from the file CH1.ISF to the REF1 internal waveform storage location.

- Path separators may be either forward slashes (/) or back slashes (\)
- The file and directory names have no more than eight characters as the base name, and no more than three characters as the extension as in 8.3 format

---

**NOTE.** Using back slash as a path separator may produce some unexpected results, depending on how your application treats escaped characters. Many applications recognize the sequence of back slash followed by an alphabetic character as an escaped character, and, as such, interpret that alphabetic character as a control character. For example, the sequence "\n" may be interpreted as a newline character; "\t" may be interpreted as a tab character. To ensure that this interpretation does not occur, you can use double back slashes. For example, "E:\\testfile.txt".

---

**Table 2-20: File System Commands**

| Command               | Description   |
|-----------------------|---|
| FILESystem            | Returns the file system state   |
| FILESystem:CWD        | Sets or returns the current working directory for FILESystem commands.        |
| FILESystem:DELEte     | Deletes a named file or directory   |
| FILESystem:DIR?       | Returns a list of directory contents  |
| FILESystem:FORMat     | Formats a named drive   |
| FILESystem:FREESpace? | Returns the number of bytes of free space on the current drive                |
| FILESystem:MKDir      | Creates a new directory   |
| FILESystem:READFile   | Writes the contents of the specified file to the specified interface          |
| FILESystem:REName     | Assigns a new name to an existing file  |
| FILESystem:RMDir      | Deletes a named directory   |
| FILESystem:WRITEFile  | Writes the specified block data to the oscilloscope current working directory |

## FilterVu Command Group

Use the commands in the FilterVu Command Group to set up the FilterVu feature. FilterVu provides a variable low pass filter to block unwanted noise while still displaying high frequency events. The oscilloscope does this by superimposing a filtered foreground waveform over a glitch capture background waveform.

**Table 2-21: FilterVuCommands**

| Command                  | Description   |
|--------------------------|---|
| DISplay:GLITch           | Controls the display of the glitch capture waveform |
| DISplay:INTENSITY:GLITch | Sets the glitch intensity on the display            |

**Table 2-21: FilterVuCommands (cont.)**

| <b>Command</b>                                | <b>Description</b>  |
|---|---|
| <a href="#">FILTERVu:FREQuency</a>            | Sets or queries the FilterVu frequency  |
| <a href="#">FILTERVu:FREQuency:AVAILable?</a> | Returns a comma separated list of filter frequencies available based on the current instrument settings |

For more information, refer to Appendix B: Waveform Data in DPO/MSO2000 Series Instruments. (See page 3-1.)

## Hard Copy Command Group

Use the commands in the Hard Copy Command Group to make hard copies.

**Table 2-22: Hard Copy Commands**

| <b>Command</b>                    | <b>Description</b>   |
|-----------------------------------|--|
| <a href="#">HARDCopy</a>          | Sends a copy of the screen display to the selected printer                         |
| <a href="#">HARDCopy:INKSaver</a> | Changes hard copy output to print color traces and graticule on a white background |
| <a href="#">HARDCopy:PREVIEW</a>  | Previews the current screen contents with the InkSaver palette applied             |



## Horizontal Command Group

Use the commands in the Horizontal Command Group to control the oscilloscope time bases. You can set the time-per-division of the main time base. You can also use the Horizontal commands to set the scale, horizontal position, and reference of the time base.

**Table 2-23: Horizontal Commands**

| Command   | Description   |
|---|---|
| <a href="#">HORizontal?</a>                           | Returns settings for the horizontal commands  |
| <a href="#">HORizontal:ACQLENGTH?</a>                 | Returns the record length   |
| <a href="#">HORizontal:DELay:MODE</a>                 | Sets or returns the horizontal delay mode   |
| <a href="#">HORizontal:DELay:TIME</a>                 | Sets or returns the horizontal delay time   |
| <a href="#">HORizontal:DIGital:RECOrdlength:MAIN?</a> | Returns the record length of the main digital acquisition   |
| <a href="#">HORizontal:DIGital:SAMPLERate:MAIN?</a>   | Returns the sample rate of the main digital acquisition   |
| <a href="#">HORizontal:POSition</a>                   | Sets or returns the horizontal position   |
| <a href="#">HORizontal:PREViewstate?</a>              | Returns whether or not the acquisition system is in the preview state   |
| <a href="#">HORizontal:RECOrdlength</a>               | Sets the horizontal record length to the number of data points in each frame<br>Or returns the current horizontal record length |
| <a href="#">HORizontal:RESOLution</a>                 | Sets or returns the horizontal record length to the number of data points in each frame   |
| <a href="#">HORizontal:SAMPLERate</a>                 | Sets or returns the current horizontal sample rate  |
| <a href="#">HORizontal:SCALE</a>                      | Sets or returns the time base horizontal scale  |
| <a href="#">HORizontal:TRIGger:POSition</a>           | Sets the horizontal position when delay mode is OFF   |

## Mark Command Group

Use the commands in the Mark Command Group to identify areas of the acquired waveform that warrant further investigation.

**Table 2-24: Mark Commands**

| Command              | Description  |
|----------------------|--|
| <a href="#">MARK</a> | Move to the next or previous mark on the waveform or returns all learnable settings from the mark commands |

Table 2-24: Mark Commands (cont.)

| Command                      | Description  |
|------------------------------|--|
| MARK:CREATE                  | Creates a mark on a particular waveform or all waveforms in a column                 |
| MARK:DELEte                  | Deletes a mark on a particular waveform, all waveforms in a column, or all marks     |
| MARK:FREE?                   | Returns how many marks are free to be used   |
| MARK:SElected:END?           | Returns the end of the selected mark, in terms of 0 to 100% of the waveform          |
| MARK:SElected:FOCUS?         | Returns the focus of the selected mark, in terms of 0 to 100% of the waveform        |
| MARK:SElected:MARKSINCOLumn? | Returns how many marks are in the current zoom pixel column                          |
| MARK:SElected:OWNer?         | Returns the owner of the selected mark   |
| MARK:SElected:SOURCE?        | Returns the source waveform of the selected mark                                     |
| MARK:SElected:START?         | Returns the start of the selected mark, in terms of 0 to 100% of the waveform        |
| MARK:SElected:STATe?         | Returns the on or off state of the selected mark                                     |
| MARK:SElected:ZOOM:POSITION? | Returns the position of the selected mark, in terms of 0 to 100% of the upper window |
| MARK:TOTal?                  | Returns how many marks are used  |

## Math Command Group

Use the commands in the Math Command Group to create and define a math waveform. Use the available math functions to define your math waveform.

The math waveform you create depends on sources listed in the math expression. If you change these sources, the math waveform you previously defined will be affected.

Math expressions can be simple without any mathematical computations. For example: CH1, which specifies that a waveform shows the signal source of Channel 1.

The acquisition of a live waveform can stop for several reasons: You can turn off the channel, stop the waveform, or stop the trigger. When you turn off the channel, math continues and data is acquired but is not displayed. When you stop either the waveform or the trigger, the math calculation stops, and the last math calculation performed is displayed.

When a live waveform update or reference waveform is altered, math waveforms containing those waveforms as sources are also updated to reflect the changes. Remember that sources must exist, but do not need to be displayed, to be used in and to update math waveforms.

**Table 2-25: Math Commands**

| Command                                     | Description   |
|---|---|
| <a href="#">MATH[1]?</a>                    | Returns the definition of the math waveform   |
| <a href="#">MATH[1]:DEFine</a>              | Sets or returns the current math function as a text string  |
| <a href="#">MATH[1]:HORizontal:POSition</a> | Sets or returns the math horizontal display position for FFT or (non-live) math reference waveforms |
| <a href="#">MATH[1]:HORizontal:SCAle</a>    | Sets or returns the math horizontal display scale for FFT or for Dual Math waveforms                |
| <a href="#">MATH[1]:HORizontal:UNIts</a>    | Returns the math waveform horizontal unit value   |
| <a href="#">{MATH MATH1}:LABel</a>          | Sets or queries the waveform label for the math waveform  |
| <a href="#">MATH[1]:SPECTral:MAG</a>        | Sets or returns the units of spectral magnification in the math string                              |
| <a href="#">MATH[1]:SPECTral:WINDow</a>     | Sets or returns the window function for math waveform spectral input data                           |
| <a href="#">MATH[1]:VERTical:POSition</a>   | Sets or returns the vertical position of the currently selected math type                           |
| <a href="#">MATH[1]:VERTical:SCAle</a>      | Sets or returns the vertical scale of the currently selected math type                              |

Table 2-25: Math Commands (cont.)

| Command                                 | Description                                 |
|---|---|
| <a href="#">MATH[1]:VERTical:UNits?</a> | Returns the math waveform vertical units    |
| <a href="#">MATH[1]:TYPE</a>            | Sets or returns the math waveform mode type |

## Measurement Command Group

Use the commands in the Measurement Command Group to control the automated measurement system.

Up to four automated measurements can be displayed on the screen. In the commands, these measurement readouts are named MEAS<x>, where <x> is the measurement number.

In addition to the four displayed measurements, the measurement commands let you specify an additional measurement, IMMEd. The immediate measurement has no front-panel equivalent. Immediate measurements are never displayed. Because they are computed only when needed, immediate measurements slow the waveform update rate less than displayed measurements.

Whether you use displayed or immediate measurements, use the VALue query to obtain measurement results.

Measurement commands can set and query measurement parameters. You can assign some parameters, such as waveform sources, differently for each measurement. Other parameters, such as reference levels, have only one value, which applies to all measurements.

Table 2-26: Measurement Commands

| Command   | Description  |
|---|--|
| <a href="#">MEASUrement?</a>                          | Returns all measurement parameters   |
| <a href="#">MEASUrement:CLEARSNapshot</a>             | Removes the measurement snapshot display   |
| <a href="#">MEASUrement:GATing</a>                    | Sets or returns the measurement gating   |
| <a href="#">MEASUrement:IMMed?</a>                    | Returns all immediate measurement setup parameters   |
| <a href="#">MEASUrement:IMMed:DELay?</a>              | Returns information about the immediate delay measurement  |
| <a href="#">MEASUrement:IMMed:DELay:DIRection</a>     | Sets or returns the search direction to use for immediate delay measurements                         |
| <a href="#">MEASUrement:IMMed:DELay:EDGE&lt;x&gt;</a> | Sets or returns the slope of the edge used for immediate delay “from” and “to” waveform measurements |
| <a href="#">MEASUrement:IMMed:SOUrce1</a>             | Sets or returns the “from” source for all single channel immediate measurements                      |

Table 2-26: Measurement Commands (cont.)

| Command                             | Description  |
|-------------------------------------|--|
| MEASUrement:IMMed:SOUrce2           | Sets or returns the source to measure “to” for phase or delay immediate measurements         |
| MEASUrement:IMMed:TYPe              | Sets or returns the type of the immediate measurement  |
| MEASUrement:IMMed:UNIts?            | Returns the units of the immediate measurement   |
| MEASUrement:IMMed:VALue?            | Returns the value of the immediate measurement   |
| MEASUrement:INDICators?             | Returns all measurement indicator parameters   |
| MEASUrement:INDICators:HORIZ<x>?    | Returns the position of the specified horizontal measurement indicator                       |
| MEASUrement:INDICators:NUMHORZ?     | Returns the number of horizontal measurement indicators currently being displayed            |
| MEASUrement:INDICators:NUMVERT?     | Returns the number of vertical measurement indicators currently being displayed              |
| MEASUrement:INDICators:STATE        | Sets or returns the state of visible measurement indicators                                  |
| MEASUrement:INDICators:VERT<x>?     | Returns the value of the specified vertical measurement indicator                            |
| MEASUrement:MEAS<x>?                | Returns all measurement parameters   |
| MEASUrement:MEAS<x>:COUNT?          | Returns the number of values accumulated since the last statistical reset                    |
| MEASUrement:MEAS<x>:DELay?          | Returns the delay measurement parameters for the specified measurement                       |
| MEASUrement:MEAS<x>:DELay:DIRection | Sets or returns the search direction to use for delay measurements                           |
| MEASUrement:MEAS<x>:DELay:EDGE<x>   | Sets or returns the slope of the edge to use for delay “from” and “to” waveform measurements |
| MEASUrement:MEAS<x>:MAXimum?        | Returns the maximum value found since the last statistical reset                             |
| MEASUrement:MEAS<x>:MEAN?           | Returns the mean value accumulated since the last statistical reset                          |
| MEASUrement:MEAS<x>:MINImum?        | Returns the minimum value found since the last statistical reset                             |
| MEASUrement:MEAS<x>:SOURCE[1]       | Sets or returns the channel from which measurements are taken                                |
| MEASUrement:MEAS<x>:SOURCE2         | Sets or returns the channel to which measurements are sent                                   |

Table 2-26: Measurement Commands (cont.)

| Command                              | Description   |
|--------------------------------------|---|
| MEASUrement:MEAS<x>:STATE            | Sets or returns whether the specified measurement slot is computed and displayed      |
| MEASUrement:MEAS<x>:STDdev?          | Returns the standard deviation of values accumulated since the last statistical reset |
| MEASUrement:MEAS<x>:TYPe             | Sets or returns the measurement<x> type   |
| MEASUrement:MEAS<x>:UNIts?           | Returns measurement<x> units  |
| MEASUrement:MEAS<x>:VALue?           | Returns the value of measurement<x>   |
| MEASUrement:METhod                   | Sets or returns the method used for calculating reference levels                      |
| MEASUrement:REFLevel?                | Returns the current reference level parameters  |
| MEASUrement:REFLevel:ABSolute:HIGH   | Sets or returns the top reference level for rise time                                 |
| MEASUrement:REFLevel:ABSolute:LOW    | Sets or returns the low reference level for rise time                                 |
| MEASUrement:REFLevel:ABSolute:MID    | Sets or returns the mid reference level for measurements                              |
| MEASUrement:REFLevel:ABSolute:MID2   | Sets or returns the mid reference level for delay "to" measurements                   |
| MEASUrement:REFLevel:METhod          | Sets or returns the method for assigning high and low reference levels                |
| MEASUrement:REFLevel:PERCent:HIGH    | Sets or returns the top reference percent level for rise time                         |
| MEASUrement:REFLevel:PERCent:LOW     | Sets or returns the low reference percent level for rise time                         |
| MEASUrement:REFLevel:PERCent:MID     | Sets or returns the mid reference percent level for waveform measurements             |
| MEASUrement:REFLevel:PERCent:MID2    | Sets or returns the mid reference percent level for second waveform measurements      |
| MEASUrement:SNAPShot                 | Displays the measurement snapshot list  |
| MEASUrement:IMMed:SOUrce<x>          | Sets or returns the source for the current single channel measurement                 |
| MEASUrement:MEAS<x>:SOUrce<x>        | Sets or returns the source for the specified measurement.                             |
| MEASUrement:REFLevel:ABSolute:MID<x> | Sets or returns the mid reference level for the specified channel in absolute volts   |
| MEASUrement:REFLevel:PERCent:MID<x>  | Sets or returns the mid reference level for the specified channel in percent          |

## Miscellaneous Command Group

Use the commands in the Miscellaneous Command Group to perform actions that do not fit into other categories.

Several commands and queries are common to all 488.2-1987 devices. The 488.2-1987 standard defines these commands. The common commands begin with an asterisk (\*) character.

**Table 2-27: Miscellaneous Commands**

| Command      | Description   |
|--------------|---|
| AUTOSet      | Sets the vertical, horizontal and trigger controls to provide a stable display of the selected waveform             |
| CLEARMenu    | Clears the current menu from the display  |
| DATE         | Sets or returns the date displayed by the oscilloscope  |
| *DDT         | Sets or returns the commands that will be executed by the group execute trigger                                     |
| FPanel:PRESS | Simulates the action of pressing a specified front-panel button   |
| FPanel:TURN  | Duplicates the action of turning a specified front-panel control knob   |
| GPIBUsb:ID?  | Returns the identification string of the connected adaptor module and firmware version                              |
| HEADer       | Sets or returns the Response Header Enable State  |
| ID?          | Returns identifying information about the oscilloscope and its firmware   |
| *IDN?        | Returns the same information as the ID? command except the data is formatted according to Tektronix Codes & Formats |
| LANGuage     | Sets or returns the user interface display language   |
| LOCK         | Sets or returns the front panel lock state  |
| *LRN?        | Returns a listing of oscilloscope settings  |
| MESSage      | Sets or queries message parameters  |
| NEWpass      | Changes the password for user protected data  |
| PASSWord     | Enables the *PUD and NEWpass set commands   |
| REM          | Specifies a comment, which is ignored by the oscilloscope   |
| SET?         | Returns a listing of oscilloscope settings  |

**Table 2-27: Miscellaneous Commands (cont.)**

| <b>Command</b>                | <b>Description</b>  |
|-------------------------------|---|
| TEKSecure                     | Initializes both waveform and setup memories  |
| TIME                          | Sets or returns the time displayed by the oscilloscope  |
| TOTALuptime?                  | Returns the total number of hours that the oscilloscope has been turned on since the nonvolatile memory was last programmed |
| *TRG                          | Performs the group execute trigger (GET)  |
| *TST?                         | Tests the interface and returns the status  |
| UNLock                        | Unlocks front panel   |
| USBTMC                        | Returns the USBTMC information  |
| USBTMC:PRODUCTID:DECimal?     | Returns the product ID in decimal format  |
| USBTMC:PRODUCTID:HEXadecimal? | Returns the product ID in hexadecimal format  |
| USBTMC:SERIALnumber?          | Returns the instrument serial number  |
| USBTMC:VENDORID:DECimal?      | Returns the vendor ID in decimal format   |
| USBTMC:VENDORID:HEXadecimal?  | Returns the vendor ID in hexadecimal format   |
| VERBose                       | Sets or returns the verbose state   |

## PictBridge Command Group

Use the commands in the PictBridge to store the printer settings.

**Table 2-28: PictBridge Commands**

| <b>Command</b>       | <b>Description</b>  |
|----------------------|---|
| PICTBridge:DATEPrint | Enables or disables printing the date on the print output                                 |
| PICTBridge:DEfault   | Sets the arguments for all PictBridge commands to their default values                    |
| PICTBridge:IDPrint   | Enables or disables printing the oscilloscope model and serial number on the print output |
| PICTBridge:IMAGESize | Sets or returns the image print size  |
| PICTBridge:PAPERSize | Sets the output print paper size  |
| PICTBridge:PAPERType | Sets or returns the paper type  |
| PICTBridge:PRINTQual | Sets or returns the output print quality  |
| USBDevice:CONFigure  | Enables or disables the rear USB port for use with Pictbridge printers                    |



## Save and Recall Command Group

Use the commands in the Save and Recall Command Group to store and retrieve internal waveforms and settings. When you save a setup, you save all the settings of the oscilloscope. When you recall a setup, the oscilloscope restores itself to the state it was in when you originally saved the setting.

**Table 2-29: Save and Recall Commands**

| Command                              | Description  |
|--------------------------------------|--|
| FACTory                              | Resets the oscilloscope to factory default settings  |
| *RCL                                 | Recalls saved oscilloscope settings  |
| RECALL:SETUp                         | Recalls saved oscilloscope settings  |
| RECALL:WAVEform                      | Recalls a stored waveform to a reference location  |
| REF<x>:SCALE                         | Sets or returns the vertical scale for the channel specified by <x>, where x is the reference channel number   |
| *SAV                                 | Stores the state of the oscilloscope to a specified memory location  |
| SAVe:ASSIgn:TYPe                     | Sets or returns the assignment of the data to be saved   |
| SAVe:EVENTable:BUS<x>                | Saves event table data from bus<x> to a specified file and location  |
| SAVe:IMAGe                           | Saves a capture of the screen image into the specified file  |
| SAVe:IMAGe:FILEFormat                | Sets or returns the file format to use for saving screen images when the file type cannot be determined from the given file name or when screen images are captured by using the front panel |
| SAVe:IMAGe:LAYout                    | Sets or returns the layout to use for saved screen images  |
| SAVe:SETUp                           | Saves the state of the oscilloscope to a specified memory location or file   |
| SAVe:WAVEform                        | Saves a waveform to one of four reference memory locations or a file   |
| SAVe:WAVEform:FILEFormat             | Sets or returns the format for saved waveforms   |
| SAVe:WAVEform:GATIng                 | Specifies whether save waveform operations should save the entire waveform or a specified portion of the waveform  |
| SAVe:WAVEform:SPREADSheet:RESOLution | Specifies whether to save the full (LRL) or reduced (thumb) resolution waveform to a CSV file.   |

**Table 2-29: Save and Recall Commands (cont.)**

| <b>Command</b> | <b>Description</b>   |
|----------------|--|
| SETUP<x>:DATE? | Returns the date when the specified oscilloscope setup was saved |
| SETUP<x>:LABEL | Sets or returns the specified oscilloscope setup label           |
| SETUP<x>:TIME? | Returns the time when the specified oscilloscope setup was saved |

## Search Command Group

Use the commands in the Search Commands Group to seek out information in waveform records.

### Search Commands

| Command   | Description   |
|---|---|
| <a href="#">SEARCH?</a>   | Returns all search-related settings   |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:COPy</a>   | Copies the search criteria to the trigger, or the trigger criteria to the search. |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:STATE</a>  | Sets the search state to on or off  |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TOTAL?</a>   | Returns the total number of matches for search <x>                                |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:CONDition</a>                   | Sets or returns the search condition for CAN search                               |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:DIRection</a>              | Sets or returns the CAN search condition to be valid on a READ, WRITE or either   |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:QUALifier</a>              | Sets or returns the CAN data qualifier  |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:SIZE</a>                   | Sets or returns the length of the data string in bytes to be used for CAN search  |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:VALue</a>                  | Sets or returns the binary data string to be used for CAN search                  |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:FRAMEType</a>                   | Sets or returns the CAN Frame Type to be used                                     |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN{:IDentifier :ADDRess}:MODE</a>  | Sets or returns the CAN addressing mode to standard or extended format            |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN{:IDentifier :ADDRess}:VALue</a> | Sets or returns the binary address string to be used for CAN search               |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:LIN:CONDition</a>                   | Sets or returns the search condition for a LIN search                             |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:LIN:DATA:HIVALue</a>                | Sets or returns the binary data string  |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:LIN:DATA:QUALifier</a>              | Sets or returns the LIN data qualifier  |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:LIN:DATA:SIZE</a>                   | Sets or returns the length of the data string in bytes                            |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:LIN:DATA:VALue</a>                  | Sets or returns the binary data string used for a LIN search                      |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:LIN:ERRTYPE</a>                     | Sets or returns the error type used for a LIN Search                              |
| <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:LIN:IDentifier:VALue</a>            | Sets or returns the binary address string used for LIN search                     |

Search Commands (cont.)

| Command  | Description   |
|--|---|
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:MODE         | Sets or returns the I2C address mode to 7 or 10-bit   |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:TYPE         | Sets or returns the I2C address type to I2C special addresses   |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:VALue        | Sets or returns the binary address string to be used for I2C search   |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:CONDition            | Sets or returns the search condition for I2C search   |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:DIRectioN       | Sets or returns the I2C search condition to be valid on a READ, WRITE or either                               |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:SIZE            | Sets or returns the length of the data string in bytes to be used for I2C search                              |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:VALue           | Sets or returns the binary data string to be used for I2C search  |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:PARAllel:VALue           | Sets or returns the binary data string to be used for a parallel trigger search                               |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:CONDition         | Sets or returns the trigger condition for a RS232 trigger   |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:SIZE      | Sets or returns the length of the data string for a RS232 trigger, if the trigger condition is RX             |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue     | Sets or returns the binary data string for a RS232 trigger, if the condition involves RX                      |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE      | Sets or returns the length of the data string to be used for a RS232 Trigger, if the Trigger condition is TX  |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue     | Sets or returns the binary data string to be used for a RS232 trigger, if the condition involves RX           |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:CONDition            | Sets or returns the search condition for SPI search   |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{MISO IN}:VALue  | Sets or returns the binary data string to be used for SPI search if the search condition is MISO or MISOMOSI. |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{MOSI OUT}:VALue | Sets or returns the binary data string for an SPI search if the search condition is MISO or MISOMOSI          |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA:SIZE            | Sets or returns the length of the data string in bytes to be used for SPI search                              |
| SEARCH:SEARCH<x>:TRIGger:A:BUS:SOUrce                        | Sets or returns the bus for a serial search   |
| SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe                        | Sets or returns the slope for an edge search  |

## Search Commands (cont.)

| Command   | Description  |
|---|--|
| SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOURCE                  | Sets or returns the source waveform for an edge search   |
| SEARCH:SEARCH<x>:TRIGger:A:LEVel                        | Sets or returns the level for an edge search   |
| SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x>                  | Sets or returns the level for an edge search of the specified channel                            |
| SEARCH:SEARCH<x>:TRIGger:A:LEVel:MATH                   | Sets or returns the math waveform level for edge search  |
| SEARCH:SEARCH<x>:TRIGger:A:LEVel:REF<x>                 | Sets or returns the reference waveform level for edge search                                     |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:FUNCTION               | Sets or returns the logic operator for the logic search  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CH<x>            | Sets or returns the Boolean logic criteria for the logic search                                  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:EDGE       | Sets or returns whether the clock edge is rise or fall for a logic search                        |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:SOURCE     | Sets or returns the clock source definition for logic search                                     |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:MATH             | Sets or returns the Boolean logic criteria for the logic search                                  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:REF<x>           | Sets or returns the Boolean logic criteria for the logic search                                  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:PATtern:INPut:CH<x>    | Sets or returns the Boolean logic criteria for the logic search                                  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:D<x>             | Sets or returns the waveform logic criteria for a trigger search                                 |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:PATtern:INPut:MATH     | Sets or returns the Boolean logic criteria for the logic search                                  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:PATtern:INPut:D<x>     | Sets or returns the logic criteria for a logic pattern search to determine where to place a mark |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:PATtern:INPut:REF<x>   | Sets or returns the Boolean logic criteria for the logic search                                  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:PATtern:WHEn           | Sets or returns the condition for generating a logic pattern search                              |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:PATtern:WHEn:LESSLimit | Sets or returns the maximum time that the selected pattern may be true                           |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:PATtern:WHEn:MORELimit | Sets or returns the minimum time that the selected pattern may be true                           |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:THReshold:CH<x>        | Sets or returns the channel threshold level for an logic search                                  |
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:THReshold:MATH         | Sets or returns the math waveform threshold level for logic search                               |

Search Commands (cont.)

| Command   | Description   |
|---|---|
| SEARCH:SEARCH<x>:TRIGger:A:LOGic:THReshold:REF<x>         | Sets or returns the reference waveform threshold level for logic search   |
| SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:CH<x>           | Sets or returns the lower waveform threshold level for all channel waveform searches  |
| SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:MATH            | Sets or returns the lower waveform threshold level for all math waveform searches   |
| SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:REF<x>          | Sets or returns the lower waveform threshold level for all reference waveform searches  |
| SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:POLarity            | Sets or returns the polarity for a pulse search   |
| SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:SOUrce              | Sets or returns the source waveform for a pulse search  |
| SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:WHEn                | Sets or returns the condition for generating a pulse width search   |
| SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:WIDth               | Sets or returns the pulse width setting for a pulse width search  |
| SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity                  | Sets or returns the polarity setting for a runt search  |
| SEARCH:SEARCH<x>:TRIGger:A:RUNT:SOUrce                    | Sets or returns the source setting for a runt search  |
| SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn                      | Sets or returns the condition setting for a runt search   |
| SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth                     | Sets or returns the width setting for a runt search   |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:EDGE             | Sets or returns the clock slope setting for a setup/hold search   |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:SOUrce           | Sets or returns the clock source setting for an setup/hold search   |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:THReshold        | Sets or returns the clock threshold setting for an setup/hold search  |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:SOUrce            | Sets or returns the data source setting for an setup/hold search  |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:THReshold         | Sets or returns the data threshold setting for an setup/hold search   |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold:CH<x>        | Sets or returns the trigger search setup and hold lower threshold to determine where to place a mark. Ch<x> is the channel number |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold{MATH}:MATH1} | Sets or returns the trigger search setup and hold lower threshold to determine where to place a mark                              |

## Search Commands (cont.)

| Command  | Description   |
|--|---|
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold:REF<x>        | Sets or returns the trigger search setup and hold lower threshold to determine where to place a mark. Ref<x> is the reference waveform number |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:HOLDTime                | Sets or returns the hold time setting for an setup/hold search  |
| SEARCH:SEARCH<x>:TRIGger:A:SETHold:SETTime                 | Sets or returns the setup time setting for an setup/hold search   |
| SEARCH:SEARCH<x>:TRIGger:A:TRANSition :RISEFall :DELTAtime | Sets or returns the transition time setting for an transition search  |
| SEARCH:SEARCH<x>:TRIGger:A:TRANSition :RISEFall :POLarity  | Sets or returns the polarity setting for an transition search   |
| SEARCH:SEARCH<x>:TRIGger:A:TRANSition :RISEFall :SOURce    | Sets or returns the source setting for an transition search   |
| SEARCH:SEARCH<x>:TRIGger:A:TRANSition :RISEFall :WHEn      | Sets or returns the condition setting for an transition search  |
| SEARCH:SEARCH<x>:TRIGger:A:TYPE                            | Sets or returns the trigger type setting for a search   |
| SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:CH<x>            | Sets or returns the waveform upper threshold level for all channel waveform searches  |
| SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:MATH             | Sets or returns the waveform upper threshold level for all math waveform searches   |
| SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:REF<x>           | Sets or returns the waveform upper threshold level for all reference waveform searches  |

## Status and Error Command Group

Use the commands in the Status and Error Command Group to determine the status of the oscilloscope and control events.

Several commands and queries used with the oscilloscope are common to all IEEE488.2 compliant devices. The IEEE Std 488.2-1987 defines these commands and queries. The common commands begin with an asterisk (\*) character.

Table 2-30: Status and Error Commands

| Command | Description   |
|---------|---|
| ALLEV?  | Returns all events and their messages                               |
| BUSY?   | Returns oscilloscope status   |
| *CLS    | Clears status   |
| DESE    | Sets or returns the bits in the Device Event Status Enable Register |

Table 2-30: Status and Error Commands (cont.)

| Command | Description  |
|---------|--|
| *ESE    | Sets or returns the bits in the Event Status Enable Register   |
| *ESR?   | Returns the contents of the Standard Event Status Register   |
| EVENT?  | Returns event code from the event queue  |
| EVMsg?  | Returns event code, message from the event queue   |
| EVQty?  | Return number of events in the event queue   |
| *OPC    | Generates the operation complete message in the standard event status register when all pending operations are finished<br>Or returns "1" when all current operations are finished |
| *PSC    | Sets or returns the power on status flag   |
| *PUD    | Sets or returns a string of protected user data  |
| *RST    | Resets the oscilloscope to factory default settings  |
| *SRE    | Sets or returns the bits in the Service Request Enable Register  |
| *STB?   | Returns the contents of the Status Byte Register   |
| *WAI    | Prevents the oscilloscope from executing further commands until all pending operations finish  |

## Trigger Command Group

Use the commands in the Trigger Command Group to control all aspects of triggering for the oscilloscope.

There are two triggers: A and B. Where appropriate, this command set has parallel construction between triggers.

You can set the A or B triggers to edge mode. Edge triggering lets you display a waveform at or near the point where the signal passes through a voltage level of your choosing.

You can also set A triggers to pulse, logic, or video modes. With pulse triggering, the oscilloscope triggers whenever it detects a pulse of a certain width or height. Logic triggering lets you logically combine the signals on one or channels. The oscilloscope then triggers when it detects a certain combination of signal levels. Video triggering enables you to trigger on the most common Standard Definition video standards.



Table 2-31: Trigger Commands

| Command  | Description   |
|--|---|
| TRIGger  | Forces a trigger event to occur   |
| TRIGger:FREQuency?                                 | Returns the available trigger frequency in hertz.                                 |
| TRIGger:A  | Sets A trigger level to 50% or returns current A trigger parameters               |
| TRIGger:A:BUS                                      | Sets or returns the serial trigger type   |
| TRIGger:A:BUS:B<x>:CAN:CONDition                   | Sets or returns the CAN condition   |
| TRIGger:A:BUS:B<x>:CAN:DATA:DIRection              | Sets or returns the CAN trigger condition to be valid on a READ, WRITE, or either |
| TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier              | Sets or returns the CAN data qualifier  |
| TRIGger:A:BUS:B<x>:CAN:DATA:SIZE                   | Sets or returns the length of the data string in bytes to be used for CAN trigger |
| TRIGger:A:BUS:B<x>:CAN:DATA:VALue                  | Sets or returns the binary data string to be used for CAN trigger                 |
| TRIGger:A:BUS:B<x>:CAN:FRAMEtype                   | Sets or returns the CAN trigger frame type  |
| TRIGger:A:BUS:B<x>:CAN{:Identifier}:ADDRess}:MODE  | Sets or returns the CAN addressing mode   |
| TRIGger:A:BUS:B<x>:CAN{:Identifier}:ADDRess}:VALue | Sets or returns the binary address string used for the CAN trigger                |
| TRIGger:A:BUS:B<x>:LIN:CONDition                   | Sets or returns the trigger condition for LIN                                     |
| TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue                | Sets or returns the binary data string to be used for LIN trigger                 |
| TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier              | Sets or returns the LIN data qualifier  |
| TRIGger:A:BUS:B<x>:LIN:DATA:SIZE                   | Sets or returns the length of the data string in bytes to be used for LIN trigger |
| TRIGger:A:BUS:B<x>:LIN:DATA:VALue                  | Sets or returns the binary data string  |
| TRIGger:A:BUS:B<x>:LIN:ERRTYPE                     | Sets or returns the error type  |

Table 2-31: Trigger Commands (cont.)

| Command                                    | Description   |
|--|---|
| TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue    | Sets or returns the binary address string used for LIN trigger                                  |
| TRIGger:A:BUS:B<x>:I2C:ADDRess:MODE        | Sets or returns the I2C address mode to 7 or 10-bit   |
| TRIGger:A:BUS:B<x>:I2C:ADDRess:TYPE        | Sets or returns the I2C address type to USER  |
| TRIGger:A:BUS:B<x>:I2C:ADDRess:VALue       | Sets or returns the binary address string used for the I2C trigger                              |
| TRIGger:A:BUS:B<x>:I2C:CONDition           | Sets or returns the trigger condition for I2C trigger   |
| TRIGger:A:BUS:B<x>:I2C:DATA:DIRection      | Sets or returns the I2C trigger condition valid on a READ, WRITE, or either                     |
| TRIGger:A:BUS:B<x>:I2C:DATA:SIZE           | Sets or returns the length of the data string in bytes to be used for I2C trigger               |
| TRIGger:A:BUS:B<x>:I2C:DATA:VALue          | Sets or returns the binary data string used for I2C triggering                                  |
| TRIGger:A:BUS:B<x>:PARAllel:VALue          | Sets or returns the binary data string to be used for a parallel trigger                        |
| TRIGger:A:BUS:B<x>:RS232C:CONDition        | Sets or returns the condition for a RS232C trigger  |
| TRIGger:A:BUS:B<x>:RS232C:RX:DATA:SIZE     | Sets or returns the length of the data string in Bytes for a RX RS232 Trigger                   |
| TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue    | Sets or returns the binary data string for a RX RS232 trigger                                   |
| TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE     | Sets or returns the length of the data string for a TX RS232 trigger                            |
| TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue    | Sets or returns the binary data string for a RS232 trigger if the trigger condition involves TX |
| TRIGger:A:BUS:B<x>:SPI:CONDition           | Sets or returns the trigger condition for SPI triggering  |
| TRIGger:A:BUS:B<x>:SPI:DATA{IN MISO}:VALue | Sets or returns the binary data string to be used for SPI trigger                               |

Table 2-31: Trigger Commands (cont.)

| Command                                       | Description  |
|---|--|
| TRIGger:A:BUS:B<x>:SPI:DATA{:OUT :MOSI}:VALue | Sets or returns the binary data string used for the SPI trigger  |
| TRIGger:A:BUS:B<x>:SPI:DATA:SIZE              | Sets or returns the length of the data string in bytes to be used for SPI trigger                      |
| TRIGger:A:BUS:SOURce                          | Sets or returns the source for a bus trigger   |
| TRIGger:A:EDGE?                               | Returns the source, coupling and source for the A edge trigger   |
| TRIGger:A:EDGE:COUPling                       | Sets or returns the type of coupling for the A edge trigger  |
| TRIGger:A:EDGE:SLOpe                          | Sets or returns the slope for the A edge trigger   |
| TRIGger:A:EDGE:SOURce                         | Sets or returns the source for the A edge trigger  |
| TRIGger:A:HOLDoff?                            | Returns the A trigger holdoff parameters   |
| TRIGger:A:HOLDoff:TIME                        | Sets or returns the A trigger holdoff time   |
| TRIGger:A:LEVel                               | Sets or returns the trigger level for the A trigger  |
| TRIGger:A:LEVel:AUXin                         | Sets or returns the trigger level for the AUXIN port   |
| TRIGger:A:LEVel:CH<x>                         | Specifies or returns the trigger level for the specified trigger channel                               |
| TRIGger:A:LEVel:D<x>                          | Sets or returns the trigger level for the specified digital channel <x>, where x is the channel number |
| TRIGger:A:LOGic?                              | Returns all A trigger logic settings   |
| TRIGger:A:LOGic:CLAss                         | Sets or returns the type of A trigger logic  |
| TRIGger:A:LOGic:FUNCTion                      | Sets or returns the logical combination of the input channels for the A logic trigger                  |
| TRIGger:A:LOGic:INPut?                        | Returns the logic input values for all channels  |

Table 2-31: Trigger Commands (cont.)

| Command                                | Description  |
|--|--|
| TRIGger:A:LOGic:INPut:CH<x>            | Specifies or returns the logic setting for the specified channel   |
| TRIGger:A:LOGic:INPut:D<x>             | Sets or returns the logic pattern for a trigger on digital channel <x>, where x is the channel number                |
| TRIGger:A:LOGic:INPut:CLOCK:EDGE       | Sets the polarity of the clock channel.  |
| TRIGger:A:LOGic:INPut:CLOCK:SOUrce     | Sets or returns the channel to use as the clock source   |
| TRIGger:A:LOGic:PATtern?               | Returns the conditions for generating an A logic pattern trigger   |
| TRIGger:A:LOGic:PATtern:DELTAtime      | Sets or returns the pattern trigger delta time value   |
| TRIGger:A:LOGic:PATtern:INPut:D<x>     | Sets or returns the A logic trigger input for the specified digital channel <x>, where x is the channel number.      |
| TRIGger:A:LOGic:PATtern:WHEn           | Sets or returns the pattern logic condition on which to trigger the oscilloscope                                     |
| TRIGger:A:LOGic:PATtern:WHEn:LESSLimit | Sets or returns the maximum time that the selected pattern may be true and still generate an A logic pattern trigger |
| TRIGger:A:LOGic:PATtern:WHEn:MORELimit | Sets or returns the minimum time that the selected pattern may be true and still generate an A logic pattern trigger |
| TRIGger:A:LOGic:THReshold:(CH<x> D<x>) | Sets or queries the logic trigger threshold voltage for the specified channel  |
| TRIGger:A:LOWerthreshold:CH<x>         | Sets or returns the lower threshold for the channel selected   |
| TRIGger:A:LOWerthreshold{:EXT}:AUX}    | Sets or returns the lower threshold for the Auxiliary input.   |
| TRIGger:A:MODE                         | Sets or returns the A trigger mode   |
| TRIGger:A:PULse?                       | Returns the A pulse trigger parameters   |

Table 2-31: Trigger Commands (cont.)

| Command                           | Description   |
|-----------------------------------|---|
| TRIGger:A:PULse:CLAss             | Sets or returns the type of pulse on which to trigger                                     |
| TRIGger:A:PULSEWIDTH?             | Returns the trigger A pulse width parameters  |
| TRIGger:A:PULSEWidth:POLarity     | Sets or returns the polarity for the A pulse width trigger                                |
| TRIGger:A:PULSEWidth:SOUrce       | Sets or returns the source for the pulse width trigger                                    |
| TRIGger:A:PULSEWidth:WHEn         | Sets or returns the criteria for width specification of pulse width trigger events        |
| TRIGger:A:PULSEWidth:WIDTh        | Sets or returns the width setting for the pulse width trigger                             |
| TRIGger:A:RUNT?                   | Returns the current A runt pulse trigger logic parameters                                 |
| TRIGger:A:RUNT:POLarity           | Sets or returns the polarity for the A pulse runt trigger                                 |
| TRIGger:A:RUNT:SOUrce             | Sets or returns the source for the A pulse trigger  |
| TRIGger:A:RUNT:WHEn               | Sets or returns the type of pulse width the trigger checks for when it uncovers a runt    |
| TRIGger:A:RUNT:WIDTh              | Sets or returns the minimum width for A pulse runt trigger                                |
| TRIGger:A:SETHold?                | Returns settings for setup and hold violation triggering                                  |
| TRIGger:A:SETHold:CLOCK?          | Returns clock edge polarity, voltage threshold and source input for setup/hold triggering |
| TRIGger:A:SETHold:CLOCK:EDGE      | Sets or returns the clock edge polarity for setup and hold triggering                     |
| TRIGger:A:SETHold:CLOCK:SOUrce    | Sets or returns the clock source for the A logic trigger setup and hold input             |
| TRIGger:A:SETHold:CLOCK:THReshold | Sets or returns the clock voltage threshold for setup and hold trigger                    |

Table 2-31: Trigger Commands (cont.)

| Command                                    | Description   |
|--|---|
| TRIGger:A:SETHold:DATA?                    | Returns the voltage threshold and data source for the setup/hold trigger                                    |
| TRIGger:A:SETHold:DATA:SOURce              | Sets or returns the data source for the setup and hold trigger  |
| TRIGger:A:SETHold:DATA:THReshold           | Sets or returns the data voltage threshold for setup and hold trigger                                       |
| TRIGger:A:SETHold:HOLDTime                 | Sets or returns the hold time for the setup and hold violation triggering                                   |
| TRIGger:A:SETHold:SETTime                  | Sets or returns the setup time for setup and hold violation triggering                                      |
| TRIGger:A:SETHold:THReshold:CH<x>          | Sets or queries the threshold for the channel   |
| TRIGger:A:UPPerthreshold:CH<x>             | Sets the upper threshold for the channel selected   |
| TRIGger:A{:TRANSition :RISEFall}?          | Returns the delta time, polarity, and both upper and lower threshold limits for the transition time trigger |
| TRIGger:A{:TRANSition :RISEFall}:DELTAtime | Sets or returns the delta time used in calculating the transition value                                     |
| TRIGger:A{:TRANSition :RISEFall}:POLarity  | Sets or returns the polarity for the A pulse transition trigger   |
| TRIGger:A{:TRANSition :RISEFall}:SOURce    | Sets or returns the source for transition trigger   |
| TRIGger:A{:TRANSition :RISEFall}:WHEn      | Sets or returns the relationship of delta time to transitioning signal                                      |
| TRIGger:A:TYPE                             | Sets or returns the type of A trigger   |
| TRIGger:A:VIDeo?                           | Returns the video parameters for the A trigger  |
| TRIGger:A:VIDeo:HOLDoff:FIELD              | Sets or returns the video trigger holdoff   |
| TRIGger:A:VIDeo:LINE                       | Sets or returns the video line number on which the oscilloscope triggers                                    |

Table 2-31: Trigger Commands (cont.)

| Command                       | Description  |
|-------------------------------|--|
| TRIGger:A:VIDeo:POLarity      | Sets or returns the polarity of the A video trigger                          |
| TRIGger:A:VIDeo:SOURce        | Sets or returns the polarity of the video trigger                            |
| TRIGger:A:VIDeo:STANdard      | Sets or returns the video standard   |
| TRIGger:A:VIDeo{:SYNC :FIELD} | Sets or returns the video field trigger                                      |
| TRIGger:EXTernal?             | Returns external trigger parameters  |
| TRIGger:EXTernal:PRObe        | Sets or returns the attenuation factor value of the external probe connector |
| TRIGger:EXTernal:YUNits?      | Returns the external trigger vertical (Y) units value                        |
| TRIGger:STATE?                | Returns the current state of the triggering system                           |

## Vertical Command Group

Use the commands in the Vertical Command Group to control the vertical setup of all live (channel) waveforms for acquisition and to display of channel, reference, and math waveforms.

Table 2-32: Vertical Commands

| Command                    | Description   |
|----------------------------|---|
| AUXin?                     | Returns auxiliary input parameters  |
| AUXin:PRObe                | Returns all information concerning the probe attached to auxiliary input                  |
| AUXin:PRObe:AUTOZero       | Sets the TekVPI probe attached to the auxiliary input to autozero                         |
| AUXin:PRObe:COMMAND        | Sets the state of the specified probe control   |
| AUXin:PRObe:DEGAUss        | Starts a degauss/autozero cycle on a TekVPI current probe attached to the auxiliary input |
| AUXin:PRObe:DEGAUss:STATE? | Returns the degauss state of the TekVPI current probe attached to the auxiliary input     |
| AUXin:PRObe:FORCEDRange    | Sets or returns the range of the TekVPI probe attached to the auxiliary input             |
| AUXin:PRObe:GAIN           | Sets or returns the gain factor of the probe that is attached to the auxiliary input      |

Table 2-32: Vertical Commands (cont.)

| Command                    | Description  |
|----------------------------|--|
| AUXin:PRObe:ID:SERnumber?  | Returns the serial number of the probe that is attached to the auxiliary input                     |
| AUXin:PRObe:ID:TYPE?       | Returns the type of probe that is attached to the auxiliary input                                  |
| AUXin:PRObe:RESistance?    | Returns the resistance of the probe that is attached to the Auxiliary input                        |
| AUXin:PRObe:SIGnal         | Sets or returns the input bypass setting on VPI probes that support input bypass                   |
| AUXin:PRObe:UNIts?         | Returns the units of measure of the probe that is attached to the auxiliary input                  |
| CH<x>?                     | Returns vertical parameters for the specified channel  |
| CH<x>:BANdwidth            | Sets or returns the bandwidth of the specified channel   |
| CH<x>:COUPling             | Sets or returns the coupling setting for the specified channel                                     |
| CH<x>:DESKew               | Sets or returns the deskew time for the specified channel  |
| CH<x>:IMPedance            | Sets or returns channel <x> input impedance  |
| CH<x>:INVert               | Sets or returns the invert function for the specified channel                                      |
| CH<x>:LABel                | Sets or returns the waveform label for channel <x>   |
| CH<x>:OFFSet               | Sets or returns the channel offset   |
| CH<x>:POSition             | Sets or returns the channel vertical position  |
| CH<x>:PRObe?               | Returns the gain, resistance, units, and ID of the probe that is attached to the specified channel |
| CH<x>:PRObe:AUTOZero       | Sets the TekVPI probe attached to the specified channel input to autozero                          |
| CH<x>:PRObe:COMMAND        | Sets the state of the specified probe control  |
| CH<x>:PRObe:DEGAUss        | Starts a degauss/autozero cycle on a TekVPI current probe attached to the specified channel input  |
| CH<x>:PRObe:DEGAUss:STATE? | Returns the state of the probe degauss   |
| CH<x>:PRObe:FORCEDRange    | Sets or returns the range on a TekVPI probe attached to the specified channel                      |
| CH<x>:PRObe:GAIN           | Sets or returns the gain factor of the probe that is attached to the specified channel             |
| CH<x>:PRObe:ID?            | Returns the type and serial number of the probe that is attached to the specified channel          |



Table 2-32: Vertical Commands (cont.)

| Command                      | Description  |
|------------------------------|--|
| CH<x>:PRObe:ID:SERnumber?    | Returns the serial number of the probe that is attached to the specified channel   |
| CH<x>:PRObe:ID:TYPE?         | Returns the type of probe that is attached to the specified channel  |
| CH<x>:PRObe:RESistance?      | Returns the resistance of the probe that is attached to the specified channel  |
| CH<x>:PRObe:SIGnal           | Sets or returns the input bypass setting of channel <x>TekVPI probe  |
| CH<x>:PRObe:UNIts?           | Returns the units of measure of the probe that is attached to the specified channel  |
| CH<x>:SCAlE                  | Sets or returns the vertical scale of the specified channel  |
| CH<x>:TERmination            | Sets or returns channel input termination  |
| CH<x>:VOLts                  | Sets or returns the vertical sensitivity for channel <x>   |
| CH<x>:YUNits                 | Sets or returns the units for the specified channel to a custom string   |
| REF<x>?                      | Returns reference waveform data for channel <x>  |
| REF<x>:DATE?                 | Returns the date that a reference waveform was stored  |
| REF<x>:HORizontal:DELay:TIMe | Sets or returns the horizontal position of the specified reference waveform in percent of the waveform that is displayed to the right of the center vertical graticule |
| REF<x>:HORizontal:SCAlE      | Sets or returns the horizontal scale for a reference waveform  |
| REF<x>:LABel                 | Sets or returns the specified reference waveform label   |
| REF<x>:POSition              | Sets or returns the vertical position for channel <x>, where x is the reference channel number   |
| REF<x>:TIMe?                 | Returns the time that a reference waveform was stored  |
| REF<x>:VERTical:POSition     | Sets or returns the vertical position of the specified reference waveform  |
| REF<x>:VERTical:SCAlE        | Sets or returns the reference waveform vertical scale in vertical units/div  |
| SElect                       | Returns information on which waveforms are on or off and which waveform is selected.   |

**Table 2-32: Vertical Commands (cont.)**

| <b>Command</b>                   | <b>Description</b>  |
|----------------------------------|---|
| <code>SElect:BUS&lt;x&gt;</code> | Turns on or off the specified bus waveform or returns whether the specified bus channel is on or off              |
| <code>SElect:CH&lt;x&gt;</code>  | Turns on or off the specified waveform or returns whether the specified channel is on or off                      |
| <code>SElect:D&lt;x&gt;</code>   | Turns on the display of the digital channel <x> and resets the acquisition. <x > is the channel number            |
| <code>SElect:CONTROI</code>      | Sets or returns the waveform that is selected as the implied recipient of channel-related commands                |
| <code>SElect:MATH[1]</code>      | Turns on or off the math waveform or returns whether the math waveform is on or off                               |
| <code>SElect:REF&lt;x&gt;</code> | Turns on or off the specified reference waveform or returns whether the specified reference waveform is on or off |

## Waveform Transfer Command Group

Use the commands in the Waveform Transfer Command Group to transfer waveform data points to and from the oscilloscope. Waveform data points are a collection of values that define a waveform. One data value usually represents one data point in the waveform record. When working with envelope waveforms, each data value is either the minimum or maximum of a min/max pair.

---

**NOTE.** The source waveform (as specified by the [DATA:SOURce](#)) must be turned on for the following queries to return a response:

[CURVe?](#)  
[DATA:COMPosition:AVAILable?](#)  
[WAVFrm?](#)  
[WFMOutpre:RECOrdlength?](#)  
[WFMOutpre:FILTErFreq?](#)  
[WFMOutpre:NR\\_Pt?](#)  
[WFMOutpre:PT\\_Fmt?](#)  
[WFMOutpre:PT\\_ORder?](#)  
[WFMOutpre:XUNit?](#)  
[WFMOutpre:XINcr?](#)  
[WFMOutpre:XZEro?](#)  
[WFMOutpre:PT\\_Off?](#)  
[WFMOutpre:YUNit?](#)  
[WFMOutpre:YMUlt?](#)  
[WFMOutpre:YOFf?](#)  
[WFMOutpre:YZEro?](#)

---

Before you transfer waveform data, you must specify the data format, composition, resolution, record length, and waveform source or destination. For more information on the waveform transfer features which are unique to the DPO/MSO2000 series, refer to Appendix B: Waveform Data in DPO/MSO2000 Series Instructions.(See page 3-1.)

## Data Formats

All data points for DPO models are signed integer format only. Valid data widths for CH1–CH4, MATH, and REF1, REF2 are 1 and 2-byte widths. The valid data widths for the digital collection (MSO models) is either 4 or 8-byte widths.

The oscilloscope can transfer waveform data in either ASCII or binary format. You specify the format with the [DATA:ENCdg](#) command.

**ASCII Data.** ASCII data is represented by signed integer values. The range of the values depends on the byte width specified. One byte wide data ranges from -128 to 127. Two byte wide data ranges from -32768 to 32767.

Each data value requires two to seven characters. This includes one to five characters to represent the value, another character, if the value is negative, to represent a minus sign, and a comma to separate the data points.

An example ASCII waveform data string may look like this:

```
CURVE<space>-110, -109, -110, -110, -109, -107, -109, -107, -106,
-105, -103, -100, -97, -90, -84, -80
```

---

**NOTE.** *You can use ASCII to obtain a readable and easier to format output than binary. However, the oscilloscope may require bytes to send the same values with ASCII than with binary, reducing transmission speed.*

*The use of ASCII for waveform data transfer is inefficient. ASCII-formatted Waveform (WAVFRM?) and Curve (CURVE?) queries, exceeding 1 M points, are not supported.*

---

**Binary Data.** Binary data is represented by signed integer or positive integer values. The range of the values depends on the byte width specified. When the byte width is one, signed integer data ranges from -128 to 127, and positive integer values range from 0 to 255. When the byte width is two, the values range from -32768 to 32767. and positive integer values range from 0 to 65,535.

**Table 2-33: Binary data ranges**

| Byte width | Signed integer range | Positive integer range |
|------------|----------------------|------------------------|
| 1          | -128 to 127          | 0 to 255               |
| 2          | 32,768 to 32,767     | 0 to 65,535            |

The defined binary formats also specify the order in which the bytes are transferred. The four binary formats are RIBinary, RPBinary, SRIBinary, and SRPBinary.

RIBinary is signed integer where the most significant byte is transferred first, and RPBinary is positive integer where the most significant byte is transferred first. SRIBinary and SRPBinary correspond to RIBinary and RPBinary respectively but use a swapped byte order where the least significant byte is transferred first. The byte order is ignored when DATA:WIDth is set to 1.

## Waveform Data and Record Lengths

You can transfer multiple points for each waveform record. You can transfer a portion of the waveform or you can transfer the entire record. You can use the DATA:START and DATA:STOP commands to specify the first and last data points of the waveform record.

When transferring data into the oscilloscope, you must first specify the record length of the destination waveform record. You do this with the WFMInpre:NR\_Pt

command. Next, specify the first data point within the waveform record. For example, when you set `DATA:START` to 1, data points will be stored starting with the first point in the record. The oscilloscope will ignore the value set by `DATA:STOP` when reading in data. It will stop reading data when there is no data to read or when it has reached the specified record length.

When transferring data from the oscilloscope, you must specify the first and last data points in the waveform record. Setting `DATA:START` to 1 and `DATA:STOP` to the record length will always return the entire waveform.

### **Waveform Data Locations and Memory Allocation**

The `DATA:SOURce` command specifies the waveform source when transferring a waveform from the oscilloscope. You can only transfer one waveform at a time. Waveforms sent to the oscilloscope are always stored in one of the reference memory locations. Use the `DATA:DESTination` command to specify a reference memory location.

### **Waveform Preamble**

Each waveform you transfer has an associated waveform preamble, which contains information such as horizontal scale, vertical scale, and the other settings in effect when the waveform was created. Refer to the individual `WFMinpre` and `WFMOupre` commands for information.

### **Scaling Waveform Data**

Once you transfer the waveform data to the controller, you can convert the data points into voltage values for analysis using information from the waveform preamble.

### **Transferring Waveform Data from the Oscilloscope**

You can transfer waveforms from the oscilloscope to an external controller using the following sequence:

1. Select the waveform source(s) using `DATA:SOURce`.
2. Specify the waveform data format using `DATA:ENCdg`.
3. Specify the number of bytes per data point using `WFMOupre:BYT_Nr`.
4. Specify the portion of the waveform that you want to transfer using `DATA:START` and `DATA:STOP`.
5. Transfer waveform preamble information using the `WFMOupre?` query.
6. Transfer waveform data from the oscilloscope using the `CURVe?` query.

## Transferring Waveform Data to the Oscilloscope

You can transfer waveforms to the oscilloscope from an external controller using the following sequence:

1. Specify the reference waveform using `DATA:DESTination`.
2. Specify the record length of the reference waveform using `WFMPre:NR_Pt`.
3. Specify the waveform data format using `WFMinpre:ENCdg`.
4. Specify the number of bytes per data point using `WFMinpre:BYT_Nr`.
5. Specify first data point in the waveform record using `DATA:START`.
6. Transfer waveform preamble information using `WFMinpre`.
7. Transfer waveform data to the oscilloscope using `CURVe`.

**Table 2-34: Waveform Transfer Commands**

| Command                                  | Description   |
|--|---|
| <code>CURVe</code>                       | The command format transfers waveform data to the oscilloscope (reference waveform specified by <code>DATA:DESTination</code> )<br>The query format transfers waveform data from oscilloscope specified by the <code>DATA:SOURce</code> command |
| <code>DATA</code>                        | Sets the format and location of the waveform data that is transferred with the <code>CURVe</code> Command<br>Or returns the format and location of the waveform data that is transferred with the <code>CURVe?</code> command                   |
| <code>DATA:COMPosition</code>            | Sets or returns the type of data that the <code>CURVe?</code> query returns   |
| <code>DATA:COMPosition:AVAILable?</code> | Lists the waveform data types that are available for return from the instrument under the current instrument settings   |
| <code>DATA:DESTination</code>            | Sets or returns the reference waveform for storing waveform data sent to the oscilloscope   |
| <code>DATA:ENCdg</code>                  | Sets or returns the format of outgoing waveform data  |
| <code>DATA:RESOLution</code>             | Sets or returns whether the <code>CURVe</code> query returns full resolution records (acquired data) or reduced resolution records (filtered/displayed data).   |
| <code>DATA:SOURce</code>                 | Sets or returns the location of waveform data transferred from the oscilloscope   |
| <code>DATA:START</code>                  | Sets or returns the starting point in waveform transfer   |

Table 2-34: Waveform Transfer Commands (cont.)

| Command                              | Description  |
|--------------------------------------|--|
| <a href="#">DATA:STOP</a>            | Sets or returns the ending data point in waveform transfer   |
| <a href="#">WAVFrm?</a>              | Returns a branch query containing waveform data in either binary or ASCII format, waveform formatting data, and the location of the waveform data source |
| <a href="#">WFMinpre?</a>            | Returns the waveform formatting specification to be applied to the next incoming CURVE command data  |
| <a href="#">WFMinpre:BIT_Nr</a>      | Sets or returns the number of bits per binary waveform point for the incoming waveform   |
| <a href="#">WFMinpre:BN_Fmt</a>      | Sets or returns the format of binary data for the incoming waveform  |
| <a href="#">WFMinpre:BYT_Nr</a>      | Sets or returns the data width for the incoming waveform   |
| <a href="#">WFMinpre:BYT_Or</a>      | Sets or returns the byte order of waveform points for the incoming waveform  |
| <a href="#">WFMinpre:COMPosition</a> | Sets and queries the type of waveform data to be transferred to the instrument with the <a href="#">CURVe?</a> command.                                  |
| <a href="#">WFMinpre:FILTERFreq</a>  | Sets or returns the byte order of waveform points for the outgoing waveform  |
| <a href="#">WFMinpre:ENCdg</a>       | Sets or returns the type of encoding for incoming waveform data  |
| <a href="#">WFMinpre:NR_Pt</a>       | Sets or returns the number of points in the incoming waveform record   |
| <a href="#">WFMinpre:PT_Fmt</a>      | Sets or returns the point format of incoming waveform data   |
| <a href="#">WFMinpre:PT_Off</a>      | This query always returns a 0  |
| <a href="#">WFMinpre:XINcr</a>       | Sets or returns the horizontal sampling interval between incoming waveform points  |
| <a href="#">WFMinpre:XUNit</a>       | Sets or returns the horizontal units of the incoming waveform  |
| <a href="#">WFMinpre:XZEro</a>       | Sets or returns the time of the first point in the incoming waveform   |
| <a href="#">WFMinpre:YMUlt</a>       | Sets or returns the vertical scale factor, per digitizing level, of the incoming waveform points   |
| <a href="#">WFMinpre:YOff</a>        | Sets or returns the vertical position of the incoming waveform in digitizing levels  |
| <a href="#">WFMinpre:YUNit</a>       | Sets or returns the vertical units of the incoming waveform  |

Table 2-34: Waveform Transfer Commands (cont.)

| Command                 | Description  |
|-------------------------|--|
| WFMInpre:YZEro          | Sets or returns the vertical offset of the incoming waveform   |
| WFMOutpre?              | Returns the waveform formatting data for the waveform specified by the <a href="#">DATA:SOURce</a> command   |
| WFMOutpre:BYT_Or        | Sets or returns which byte of binary waveform data is transmitted first, during a waveform data transfer, when data points require more than one byte. |
| WFMOutpre:BIT_Nr        | Sets or returns the number of bits per waveform point that outgoing waveforms contain  |
| WFMOutpre:BN_Fmt        | Sets or returns the format of binary data for the outgoing waveform  |
| WFMOutpre:BYT_Nr        | Sets or returns the data width for the outgoing waveform   |
| WFMOutpre:COMPosition   | Controls or indicates the type of data used by the <a href="#">CURVe?</a> query.   |
| WFMOutpre:ENCdg         | Sets or returns the type of encoding for outgoing waveforms  |
| WFMOutpre:FILTERFreq?   | Specifies or returns the FilterVu low pass filter frequency, which was applied to the waveform being sent to the instrument.                           |
| WFMOutpre:NR_Pt?        | Returns the number of points for the waveform transmitted in response to a <a href="#">CURVe?</a> query  |
| WFMOutpre:PT_Fmt?       | Returns the point format for the outgoing waveform   |
| WFMOutpre:PT_Off?       | This query always returns a 0 if the waveform specified by <a href="#">DATA:SOURce</a> is on or displayed  |
| WFMOutpre:PT_ORder?     | This query always returns LINEAR.  |
| WFMOutpre:RECOrdlength? | Returns the record length for the source waveform as specified by the <a href="#">DATA:SOURce</a> command.   |
| WFMOutpre:WFId?         | Returns a string describing the acquisition parameters for the outgoing waveform   |
| WFMOutpre:XINcr?        | Returns the horizontal sampling interval for the outgoing waveform   |
| WFMOutpre:XUNit?        | Returns the horizontal units for the outgoing waveform   |
| WFMOutpre:XZEro?        | Returns the time of the first point in the outgoing waveform   |



Table 2-34: Waveform Transfer Commands (cont.)

| Command          | Description  |
|------------------|--|
| WFMOutpre:YMUit? | Returns the vertical scale factor per digitizing level for the outgoing waveform |
| WFMOutpre:YOFf?  | Returns the vertical position in digitizing levels for the outgoing waveform     |
| WFMOutpre:YUNit? | Returns the vertical units for the outgoing waveform                             |
| WFMOutpre:YZEro? | Returns the vertical offset for the outgoing waveform                            |

For more information, refer to Appendix B: Waveform Data in DPO/MSO2000 Series Instructions.(See page 3-1.)

## Zoom Command Group

Use the commands in the Zoom Command Group to expand and position the waveform display horizontally and vertically, without changing the time base.

Table 2-35: Zoom Commands

| Command                        | Description   |
|--------------------------------|---|
| ZOOM?                          | Returns the current horizontal positioning and scaling of the display                         |
| ZOOM{:MODE :STATE}             | Sets or returns the zoom mode   |
| ZOOM:ZOOM1?                    | Returns the current horizontal positioning and scaling of the display                         |
| ZOOM:ZOOM1:FACtor?             | Returns the zoom factor of a particular zoom box  |
| ZOOM:ZOOM1:HORizontal:POSition | Sets or returns the horizontal zoom position for the specified waveform in the specified zoom |
| ZOOM:ZOOM1:HORizontal:SCAle    | Sets or returns the horizontal zoom scale of the specified waveform in the specified zoom     |
| ZOOM:ZOOM1:POSition            | Sets or returns the horizontal zoom position for the specified waveform in the specified zoom |
| ZOOM:ZOOM1:SCAle               | Sets or returns the horizontal zoom scale of the specified waveform in the specified zoom     |
| ZOOM:ZOOM1:STATE               | Specifies or returns a trace as zoomed, on or off   |



---

# Commands Listed in Alphabetical Order

## ACQUIRE? (Query Only)

Returns the following current acquisition parameters:

- Stop after
- Acquisition state
- Mode
- Number of averages
- Sampling mode

**Group** Acquisition

**Syntax** ACQUIRE?

**Related Commands** [ACQUIRE:MODE](#), [ACQUIRE:NUMACQ?](#), [ACQUIRE:NUMAVG](#), [ACQUIRE:STOPAFTER](#)

**Examples** ACQUIRE? might return ACQUIRE:STOPAFTER RUNSTOP;STATE 1;MODE SAMPLE;NUMAVG 16;SAMPLINGMODE RT

## ACQUIRE:MAXSAMPLERATE? (Query Only)

Returns the maximum real-time sample rate, which is the same across all DPO/MSO2000 series models.

**Group** Acquisition

**Syntax** ACQUIRE:MAXSAMPLERATE?

**Examples** ACQUIRE:MAXSAMPLERATE? might return 1.0000E+9 in a DPO2024 instrument indicating that the maximum real-time sample rate is 1 GS/s.

## ACQUIRE:MODE

Sets or returns the acquisition mode of the oscilloscope for all live waveforms.

Waveforms are the displayed data point values taken from acquisition intervals. Each acquisition interval represents a time duration set by the horizontal scale (time per division). The oscilloscope sampling system always samples at the maximum rate, so the acquisition interval may include than one sample.

The acquisition mode (which you set using this ACQUIRE:MODE command) determines how the final value of the acquisition interval is generated from the many data samples.

**Group** Acquisition

**Syntax** ACQUIRE:MODE {SAMPLE|AVERAGE}  
ACQUIRE:MODE?

**Related Commands** [ACQUIRE:NUMAVG](#), [CURVE](#)

**Arguments** SAMPLE specifies that the displayed data point value is the first sampled value that is taken during the acquisition interval. In sample mode, all waveform data has 8 bits of precision. You can request 16 bit data with a [CURVE](#) query but the lower-order 8 bits of data will be zero. SAMPLE is the default mode.

AVERAGE specifies averaging mode, in which the resulting waveform shows an average of SAMPLE data points from several separate waveform acquisitions. The oscilloscope processes the number of waveforms you specify into the acquired waveform, creating a running exponential average of the input signal. The number of waveform acquisitions that go into making up the average waveform is set or queried using the ACQUIRE:NUMAVG command.

**Examples** ACQUIRE:MODE AVERAGE sets an average acquisition mode indicating that the resulting waveform is the average of the specified number of waveform acquisitions.

ACQUIRE:MODE? might return: ACQUIRE:MODE AVERAGE indicating that the displayed waveform is the average of the specified number of waveform acquisitions.

## ACQUIRE:NUMACQ? (Query Only)

The acquisition number will continue to increase while acquisitions are running until there is a reset. Starting and stopping acquisitions do not cause this number to reset. For example, if acquisitions are running, the acquisition count will increase (assuming the oscilloscope is triggering). If you stop the acquisitions, the acquisition number will freeze at a given number (For example: 5000). If you

start acquisitions again, it will continue from 5000. The number will reset to 0 only if you change the horizontal scale while acquisitions are running.

**Group** Acquisition

**Syntax** ACQuire:NUMAcq?

**Related Commands** [ACQuire:STATE](#)

**Returns** ACQuire:NUMAcq? might return :ACQUIRE:NUMACQ 350 indicating that 350 acquisitions have occurred.

## ACQuire:NUMAVg

Sets or returns the number of waveform acquisitions that make up an averaged waveform. Use the [ACQuire:MODE](#) command to enable the Average mode. Sending this command is equivalent to turning a multipurpose knob to enter the number of waveform acquisitions to average.

**Group** Acquisition

**Syntax** ACQuire:NUMAVg <NR1>  
ACQuire:NUMAVg?

**Related Commands** [ACQuire:MODE](#)

**Arguments** <NR1> is the number of waveform acquisitions to average. The range of values is from 2 to 512 in powers of two.

**Examples** ACQUIRE:NUMAVG 16 specifies that 16 waveform averages will be performed before exponential averaging starts.

ACQUIRE:NUMAVG? might return :ACQUIRE:NUMAVG 64 indicating that there are 64 acquisitions specified for averaging.

## ACQuire:STATE

Starts or stops acquisitions. When state is set to ON or RUN, a new acquisition will be started. If the last acquisition was a single acquisition sequence, a new

single sequence acquisition will be started. If the last acquisition was continuous, a new continuous acquisition will be started.

If RUN is issued in the middle of completing a single sequence acquisition (for example, averaging or enveloping), the acquisition sequence is restarted, and any accumulated data is discarded. Also, the oscilloscope resets the number of acquisitions. If the RUN argument is issued while in continuous mode, acquisition continues.

**Group** Acquisition

**Syntax** ACQUIRE:STATE {OFF|ON|RUN|STOP|<NR1>}  
ACQUIRE:STATE?

**Related Commands** [ACQUIRE:STOPAfter](#)

**Arguments** OFF stops acquisitions.  
STOP stops acquisitions.  
ON starts acquisitions.  
RUN starts acquisitions.  
<NR1> = 0 stops acquisitions; any other value starts acquisitions.

**Examples** ACQUIRE:STATE RUN starts the acquisition of waveform data.  
ACQUIRE:STATE? might return:ACQUIRE:STATE 0 indicating that the acquisition is stopped.

## ACQUIRE:STOPAfter

Sets or returns whether the oscilloscope continually acquires acquisitions or acquires a single sequence.

**Group** Acquisition

**Syntax** ACQUIRE:STOPAfter {RUNSTOP|SEQUENCE}  
ACQUIRE:STOPAfter?

**Related Commands** [ACQUIRE:STATE](#)

|                  |  |
|------------------|--|
| <b>Arguments</b> | <p>RUNSTOP specifies that the oscilloscope will continually acquire data, if <a href="#">ACQUIRE:STATE</a> is turned on.</p> <p>SEQUENCE specifies that the next acquisition will be a single-sequence acquisition.</p>                                |
| <b>Examples</b>  | <p>ACQUIRE:STOPAFTER RUNSTOP sets the oscilloscope to continually acquire data.</p> <p>ACQUIRE:STOPAFTER? might return:ACQUIRE:STOPAFTER SEQUENCE indicating that the next acquisition the oscilloscope makes will be of the single-sequence type.</p> |

## Alias

Sets or returns the state of alias functionality. Use Alias commands to define new commands as a sequence of standard commands. You may find this useful when repeatedly using the same commands to perform certain tasks like setting up measurements. Aliases are similar to macros but do not include the capability to substitute parameters into alias bodies.

To use Alias commands, first define the alias, then turn on the alias state.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Alias  |
| <b>Syntax</b>           | <p>ALIAS {OFF ON &lt;NR1&gt;}</p> <p>ALIAS?</p>  |
| <b>Related Commands</b> | <p><a href="#">ALIAS:DEFINE</a></p> <p><a href="#">ALIAS[:STATE]</a></p>   |
| <b>Arguments</b>        | <p>OFF turns alias expansion off. If a defined alias is sent when ALIAS is off, a command error (110) will be generated.</p> <p>ON turns alias expansion on. When a defined alias is received, the specified command sequence is substituted for the alias and executed.</p> <p>&lt;NR1&gt; = 0 disables alias mode; any other value enables alias mode.</p> |
| <b>Examples</b>         | <p>ALIAS ON turns the alias feature on. When a defined alias is received, the specified command sequence is substituted for the alias and executed.</p> <p>ALIAS? might return :ALIAS 1 indicating that the alias feature is on.</p>   |

## ALias:CATalog? (Query Only)

Returns a list of the currently defined alias labels, separated by commas. If no aliases are defined, the query returns the string "".

|                 |   |
|-----------------|---|
| <b>Group</b>    | Alias   |
| <b>Syntax</b>   | ALias:CATalog?  |
| <b>Examples</b> | ALIAS:CATALOG? might return the string :ALIAS:CATALOG "SETUP1", "TESTMENU1", "DEFAULT" showing that there are three aliases named SETUP1, TESTMENU1, and DEFAULT. |

## ALias:DEFine

Assigns a sequence of program messages to an alias label. These messages are then substituted for the alias whenever it is received as a command or query, provided that ALias:STATE has been turned on. The query form of this command returns the definitions of a selected alias.

---

**NOTE.** *Attempting to give two aliases the same name causes an error. To give a new alias the name of an existing alias, the existing alias must first be deleted.*

---

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Alias   |
| <b>Syntax</b>           | ALias:DEFine <QString><,>{<QString> <Block>}<br>ALias:DEFine? <QString>   |
| <b>Related Commands</b> | <a href="#">ALias[:STATE]</a>   |
| <b>Arguments</b>        | <p>The first &lt;QString&gt; is the alias label.</p> <p>This label cannot be a command name. Labels must start with a letter and can contain only letters, numbers, and underscores; other characters are not allowed. The label must be less than or equal to 12 characters.</p> <p>The second&lt;QString&gt; or &lt;Block&gt; is a complete sequence of program messages.</p> <p>The messages can contain only valid commands that must be separated by semicolons and must follow all rules for concatenating commands. The sequence must be less than or equal to 256 characters.</p> |



**Examples** ALIAS:DEFINE "ST1",":RECALL:SETUP 5;:AUTOSET EXECUTE;:SELECT:CH1 ON" defines an alias named "ST1" that sets up the oscilloscope.

ALIAS:DEFINE? "ST1" returns :ALIAS:DEFINE "ST1",#246 :RECALL:SETUP 5;:AUTOSET EXECUTE;:SELECT:CH1 ON

## ALias:DELEte (No Query Form)

Removes a specified alias and is identical to ALias:DELEte:NAME. An error message is generated if the named alias does not exist.

**Group** Alias

**Syntax** ALIas:DELEte <QString>

**Related Commands** [\\*ESR?](#), [ALias:DELEte:ALL](#)

**Arguments** <QString> is the name of the alias to be removed. Using ALIas:DELEte without specifying an alias causes an execution error. <QString> must be a previously defined value.

**Examples** ALIAS:DELETE "SETUP1" deletes the alias named SETUP1.

## ALias:DELEte:ALL (No Query Form)

Deletes all existing aliases.

**Group** Alias

**Syntax** ALIas:DELEte:ALL

**Related Commands** [ALias:DELEte](#), [ALias:DELEte\[:NAME\]](#)

**Examples** ALIAS:DELETE:ALL deletes all existing aliases.

## ALias:DELEte[:NAME] (No Query Form)

Removes a specified alias. This command is identical to [ALias:DELEte](#)

**Group** Alias

**Syntax** ALias:DELEte[:NAME] <QString>

**Arguments** <QString> is the name of the alias to remove. Using ALias:DELEte[:NAME] without specifying an alias causes an execution error. <QString> must be an existing alias.

**Examples** ALIAS:DELETE[:NAME] "STARTUP" deletes the alias named STARTUP.

## ALias[:STATE]

Turns aliases on or off. This command is identical to the [ALias](#) command.

**Group** Alias

**Syntax** ALias[:STATE] {<NR1>|OFF|ON}  
ALias[:STATE]?

**Arguments** OFF or <NR1> = 0 turns alias expansion off. If a defined alias is sent when ALias:STATE is OFF, a command error (102) is generated.

ON or <NR1>0 turns alias expansion on. When a defined alias is received, the specified command sequence is substituted for the alias and executed.

**Examples** ALIAS[:STATE] OFF turns the command alias feature off.

ALIAS[:STATE]? returns 0 when the alias feature is off.

## ALLEv? (Query Only)

Prompts the oscilloscope to return all events and their messages (delimited by commas), and removes the returned events from the Event Queue. Use the \*ESR? query to enable the events to be returned. This command is similar to repeatedly sending \*EVMsg? queries to the oscilloscope.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Status and Error   |
| <b>Syntax</b>           | ALLEV?   |
| <b>Related Commands</b> | <a href="#">*ESR?</a> , <a href="#">EVMsg?</a>   |
| <b>Examples</b>         | ALLEV? might return :ALLEV 2225,"Measurement error, No waveform to measure; "420,"Query UNTERMINATED;" |

## AUTOSet (No Query Form)

Sets the vertical, horizontal, and trigger controls of the oscilloscope to automatically acquire and display the selected waveform.

|                  |  |
|------------------|--|
| <b>Group</b>     | Miscellaneous  |
| <b>Syntax</b>    | AUTOSet {EXECute UNDo}   |
| <b>Arguments</b> | EXECute autosets the displayed waveform.<br>UNDo restores the oscilloscope settings to those present prior to the autoseg execution.       |
| <b>Examples</b>  | AUTOSET EXECUTE vertical, horizontal, and trigger controls of the oscilloscope to automatically acquire and display the selected waveform. |

## AUTOSet:ENAbLe

Enables or disables the autoseg.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Miscellaneous                              |
| <b>Syntax</b>           | AUTOSet:ENAbLe {OFF ON}<br>AUTOSet:ENAbLe? |
| <b>Related Commands</b> | <a href="#">AUTOSet</a>                    |

- Arguments** ON enables the autosegment feature.  
OFF disables the autosegment feature.
- Examples** AUTOSET:ENABLE ? might return AUTOSET:ENABLE 1 indicating that autosegment is enabled.  
AUTOSET:ENABLE OFF disables autosegment.

## AUXin? (Query Only)

Returns all auxiliary input parameters.

**Group** Vertical

**Syntax** AUXin?

## AUXin:PRObe

Returns all information concerning the probe attached to auxiliary input.

**Group** Vertical

**Syntax** AUXin:PRObe  
AUXin:PRObe?

**Examples** AUXIN:PROBE? might return AUXIN:PROBE:ID:TYPE "No Probe Detected";SERNUMBER "" ;:AUXIN:PROBE:UNITS "" ;RESISTANCE 1.0000E+6 giving information about the probe attached to the AUX In input.

## AUXin:PRObe:AUTOZero (No Query Form)

Sets the TekVPI probe attached to the Aux In input to autozero. The oscilloscope will ignore this command if the Auxiliary input does not have a TekVPI probe connected to it.

**Group** Vertical

**Syntax** AUXin:PRObe:AUTOZero {EXECute}

**Arguments** EXECute sets the probe to autozero.

**Examples** AUXin:PROBE:AUTOZERO EXECUTE

## AUXin:PRObe:COMMAND (No Query Form)

Sets the state of the probe control specified with the first argument to the state specified with the second argument. The commands and states are unique to the attached probe type. Only certain VPI probes support this command. See the probe documentation for how to set these string arguments.

**Group** Vertical

**Syntax** AUXin:PRObe:COMMAND <QString>, <QString>

**Arguments** <QString> are quoted strings specifying the probe command and value to set in the probe attached to the auxiliary input.

**Examples** AUXIN:PROBE:COMMAND "OUTPUT", "ON" turns the output of a Tektronix VPI-DPG probe on.

AUXIN:PROBE:COMMAND "MODE", "4-4V1MHZ" sets a Tektronix VPI-DPG probe to the 4-4V1MHz mode.

AUXIN:PROBE:COMMAND?"MODE" might return AUXIN:PROBE:COMMAND "MODE", "4-4V1MHZ".

## AUXin:PRObe:DEGAUss (No Query Form)

Starts a degauss/autozero cycle on a TekVPI current probe attached to the Aux In input. If you send this command to a probe that does not support this function, it is ignored

**Group** Vertical

**Syntax** AUXin:PRObe:DEGAUSS {EXECute}

**Arguments** EXECute starts a probe degauss cycle.

**Examples**    `AUXin:PROBE:DEGAUSS EXECUTE` degausses the probe attached to the Aux In input.

## **AUXin:PRObe:DEGAUss:STATE? (Query Only)**

Returns the state of the probe degauss (NEEDED, RECOMMENDED, PASSED, FAILED, RUNNING). The command will return PASSED for probes that do not support degauss operations.

**Group**    Vertical

**Syntax**    `AUXin:PRObe:DEGAUss:STATE?`

**Examples**    `AUXin:PROBE:DEGAUSS:STATE?` might return:  
`AUXin:PROBE:DEGAUSS:STATE PASSED` indicating that the probe has been degaussed.

## **AUXin:PRObe:FORCEDRange**

Changes or returns the range on a TekVPI probe attached to the Aux In input.

**Group**    Vertical

**Syntax**    `AUXin:PRObe:FORCEDRange <NR3>`  
`AUXin:PRObe:FORCEDRange?`

**Arguments**    `<NR3>` is the probe range, which is probe dependent.

## **AUXin:PRObe:GAIN**

Sets or returns the gain factor of a probe that is attached to the Aux In input.

**Group**    Vertical

**Syntax**    `AUXin:PRObe:GAIN <NR3>`  
`AUXin:PRObe:GAIN?`

**Arguments**    `<NR3>` is the probe gain, which is probe dependent.

**Examples** `AUXin:PROBE:GAIN?` might return `:AUXin:PROBE:GAIN 100.0000E-3` indicating that the attached 10x probe delivers 0.1 V to the Aux In BNC for every 1.0 V applied to the probe input.

### **AUXin:PRObe:ID:SERnumber? (Query Only)**

Returns the serial number of the probe that is attached to the auxiliary input.

**Group** Vertical

**Syntax** `AUXin:PRObe:ID:SERnumber?`

### **AUXin:PRObe:ID:TYPE? (Query Only)**

Returns the type of probe that is attached to the auxiliary input.

**Group** Vertical

**Syntax** `AUXin:PRObe:ID:TYPE?`

### **AUXin:PRObe:RESistance? (Query Only)**

Returns the resistance of the probe attached to the front panel Aux In connector.

**Group** Vertical

**Syntax** `AUXin:PRObe:RESistance?`

**Examples** `AUXin:PRObe:RESistance?` might return `:AUXin:PROBE:RESISTANCE 1.0000E+6` indicating that the input resistance of the probe attached to the front panel Aux In connector is 1 M $\Omega$ .

---

**NOTE.** *This query will return 0.0 if no probe is attached or the attached probe does not report the input resistance.*

---

## AUXin:PRObe:SIGnal

This command changes the input bypass setting on VPI probes that support input bypass, for example the TCP0001. If sent to a probe that does not support input bypass, it is ignored.

**Group** Vertical

**Syntax** AUXin:PRObe:SIGnal {BYPass|PASS}  
AUXin:PRObe:SIGnal?

**Arguments** BYPass sets the probe to Bypass mode.  
PASS sets the probe to Pass mode.

## AUXin:PRObe:UNIts? (Query Only)

Returns a string describing the units of measure of the probe attached to the Aux In input.

**Group** Vertical

**Syntax** AUXin:PRObe:UNIts?

**Examples** AUXin:PROBE:UNITS? might return: :AUXin:PROBE:UNITS "v" indicating that the units of measure for the attached probe are volts.

## BUS

Sets or returns the parameters for each bus. These parameters affect either the Serial Trigger Setup or the Bus Display.

**Conditions** This command requires a DPO2AUTO, DPO2EMBD, or DPO2COMP application module.

**Group** Bus

**Syntax** BUS



## BUS:B<x>:CAN:BITRate

Sets or returns the bit rate for CAN bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.                         |
| <b>Group</b>      | Bus  |
| <b>Syntax</b>     | BUS:B<x>:CAN:BITRate <NR1><br>BUS:B<x>:CAN:BITRate?                          |
| <b>Arguments</b>  | <NR1> sets the bit rate to the closest bit rate supported by the instrument. |
| <b>Returns</b>    | The query always returns the numerical bit rate value.                       |

## BUS:B<x>:CAN:PRObe

Sets or returns the probing method to probe CAN bus <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | BUS:B<x>:CAN:PRObe {CANH CANL RX TX DIFFerential}<br>BUS:B<x>:CAN:PRObe?  |
| <b>Arguments</b>  | CANH specifies the single-ended CANH signal, as specified by the CAN standard.<br>CANL specifies the single-ended CANL signal, as specified by the CAN standard.<br>RX specifies the receive signal on the bus side of the CAN transceiver.<br>TX specifies the transmit signal.<br>DIFFerential specifies the differential CAN signal. |

## BUS:B<x>:CAN:SAMPLEpoint

Sets or returns the sampling point during each bit period for bus <x>, where x is the bus number

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.                            |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | BUS:B<x>:CAN:SAMPLEpoint <NR1><br>BUS:B<x>:CAN:SAMPLEpoint?                     |
| <b>Arguments</b>  | <NR1> is the sample point in percent. Values are limited to 25, 30, ... 70, 75. |

## BUS:B<x>:CAN:SOURce

Sets or returns the CAN bus data source for bus <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | BUS:B<x>:CAN:SOURce {CH1 CH2 CH3 CH4 <br>D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15}<br>BUS:B<x>:CAN:SOURce? |
| <b>Arguments</b>  | CH1–CH4 is the analog channel to use as the data source.<br>D0–D15 is the digital channel to use as the data source.    |

## BUS:B<x>:DISPlay:FORMAt

Sets or returns the display format for the numerical information in the bus waveform <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO, DPO2EMBD, or DPO2COMP application module.         |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | BUS:B<x>:DISPlay:FORMAt {BINary HEXadecima ASCII MIXed}<br>BUS:B<x>:DISPlay:FORMAt? |

**Related Commands** [BUS:B<x>:TYPE](#)

**Arguments** **BiNary** specifies a binary data display.  
**HEXadecimal** specifies a hexadecimal data display.  
**ASCI**I specifies an ASCII format for RS232 only.  
**MIXed** specifies a mixed format for LIN only.

## BUS:B<x>:I2C:ADDRESS:RWINCLUDE

Sets and returns whether the read/write bit is included in the address.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:I2C:ADDRESS:RWINCLUDE {<NR1>|OFF|ON}`  
`BUS:B<x>:I2C:ADDRESS:RWINCLUDE?`

**Arguments** `<NR1> = 0` does not include the read/write bit in the address; any other value includes the read/write bit in the address.  
**OFF** does not include the read/write bit in the address.  
**ON** includes the read/write bit in the address.

**Examples** `BUS:B1:I2C:ADDRESS:RWINCLUDE ON` includes the read/write bit in the address.  
`BUS:B1:I2C:ADDRESS:RWINCLUDE?` might return  
`BUS:B1:I2C:ADDRESS:RWINCLUDE 0` indicating that the read/write bit is not included in the address.

## BUS:B<x>:I2C{:CLOCK[:SCLK]}:SOURCE

Sets or returns the I2C SCLK source for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:I2C{ :CLOCK| :SCLK}:SOURCE {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:I2C{ :CLOCK| :SCLK}:SOURCE?`

**Arguments** CH1–CH4 specifies the analog channel to use as the I2C SCLK source.  
D0–D15 specifies the digital channel to use as the I2C SCLK source.

## **BUS:B<x>:I2C{:DATA|:SDATA}:SOURCE**

Sets or returns the I2C SDATA source for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:I2C{:DATA|:SDATA}:SOURCE {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:I2C{:DATA|:SDATA}:SOURCE?`

**Arguments** CH1–CH4 specifies the analog channel to use as the I2C SDATA source.  
D0–D15 specifies the digital channel to use as the I2C SDATA source.

## **BUS:B<x>:LABEL**

Sets or returns the waveform label for bus <x>, where x is the bus number 1 through 4.

**Group** Bus

**Syntax** `BUS:B<x>:LABEL <Qstring>`  
`BUS:B<x>:LABEL?`

**Arguments** <Qstring> is an alpha-numeric string of text, enclosed in quotes, that contains the text label information for bus <x>. The text string is limited to 30 characters.

## **BUS:B<x>:LIN:BITRate**

Sets or returns the bit rate for LIN.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | <code>BUS:B&lt;x&gt;:LIN:BITRate &lt;NR1&gt;</code><br><code>BUS:B&lt;x&gt;:LIN:BITRate?</code>   |
| <b>Arguments</b>  | <NR1> is the LIN bit rate.  |
| <b>Examples</b>   | <code>BUS:B1:LIN:BITRATE 9600</code> sets the bit rate 9600.<br><code>BUS:B1:LIN:BITRATE?</code> might return <code>BUS:B1:LIN:BITRATE 2400</code> indicating that the bit rate is set to 2400. |

## BUS:B<x>:LIN:IDFORmat

Sets or returns the LIN ID format.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | <code>BUS:B&lt;x&gt;:LIN:IDFORmat {NOPARity PARity}</code><br><code>BUS:B&lt;x&gt;:LIN:IDFORmat?</code>   |
| <b>Arguments</b>  | <code>NOPARity</code> sets the LIN ID format to no parity.<br><code>PARity</code> sets the LIN ID format to parity.   |
| <b>Examples</b>   | <code>BUS:B1:LIN:IDFORMAT PARITY</code> sets the LIN ID format to parity.<br><code>BUS:B1:LIN:IDFORMAT?</code> might return <code>BUS:B1:LIN:IDFORMAT NOPARITY</code> indicating that the LIN ID format is no parity. |

## BUS:B<x>:LIN:POLARity

Sets or returns the LIN polarity.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module. |
|-------------------|--|

|                  |   |
|------------------|---|
| <b>Group</b>     | Bus   |
| <b>Syntax</b>    | BUS:B<x>:LIN:POLARity {NORMal INVerted}<br>BUS:B<x>:LIN:POLARity?   |
| <b>Arguments</b> | NORMal specifies normal LIN polarity.<br>INVerted specifies inverted LIN polarity.  |
| <b>Examples</b>  | BUS:B1:LIN:POLARITY INVERTED sets the LIN polarity to INVERTED.<br>BUS:B1:LIN:POLARITY? might return BUS:B1:LIN:POLARITY NORMAL indicating that the LIN polarity is normal. |

## BUS:B<x>:LIN:SAMPLEpoint

Sets or returns the sample point (in %) at which to sample during each bit period.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module. |
|-------------------|--|

|                  |  |
|------------------|--|
| <b>Group</b>     | Bus  |
| <b>Syntax</b>    | BUS:B<x>:LIN:SAMPLEpoint <NR1><br>BUS:B<x>:LIN:SAMPLEpoint?  |
| <b>Arguments</b> | <NR1> is the sample point (in %) at which to sample during each bit period.  |
| <b>Examples</b>  | BUS:B1:LIN:SAMPLEPOINT 10 sets the sample point is at 10% of the bit period<br>BUS:B1:LIN:SAMPLEPOINT? might return BUS:B1:LIN:SAMPLEPOINT 50 indicating that the sample point is at 50% of the bit period |

## BUS:B<x>:LIN:SOURce

Sets or returns the LIN data source.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module. |
|-------------------|--|

|              |     |
|--------------|-----|
| <b>Group</b> | Bus |
|--------------|-----|

**Syntax** `BUS:B<x>:LIN:SOURce {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:LIN:SOURce?`

**Arguments** `CH<x>` specifies the LIN source channel where x is 1 to 4.  
`D<x>` specifies the LIN source channel where x is 0 to 15.

**Examples** `BUS:B1:LIN:SOURCE CH4` sets the LIN source to channel 4.  
`BUS:B1:LIN:SOURCE?` might return `BUS:B1:LIN:SOURCE CH1` indicating that the LIN source is channel 1.

## BUS:B<x>:LIN:STANDARD

Sets or returns the LIN standard.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** `BUS:B<x>:LIN:STANDARD {V1X|V2X|MIXed}`  
`BUS:B<x>:LIN:STANDARD?`

**Arguments** `V1X` sets the LIN standard to V1X.  
`V2X` sets the LIN standard to V2X  
`MIXed` sets the LIN standard to MIXED.

**Examples** `BUS:B1:LIN:STANDARD V1X` sets the LIN standard is V1X.  
`BUS:B1:LIN:STANDARD?` might return `BUS:B1:LIN:STANDARD V2X` indicating that the LIN standard is V2X.

## BUS:B<x>:PARAllel:BIT<x>:SOURce

Sets or returns the parallel bit source for B<x>, where x is the bus number and Bit<x> is the bit number.

**Group** Bus

**Syntax**    `BUS:B<x>:PARAllel:BIT<x>:SOURce {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:PARAllel:BIT<x>:SOURce?`

**Arguments**    CH1–CH4 specifies the bit source B<x>. BIT<x> specifies the bit number.  
D0–D15 specifies the bit source B<x>. BIT<x> specifies the bit number.

## **BUS:B<x>:PARAllel:CLOCK:EDGE**

Sets or returns the parallel clock edge for bus <x>, where x is the bus number.

**Group**    Bus

**Syntax**    `BUS:B<x>:PARAllel:CLOCK:EDGE {EITHer|RISing|FALling}`  
`BUS:B<x>:PARAllel:CLOCK:EDGE?`

**Arguments**    EITHer specifies either rising or falling edge as the clock edge.  
RISing specifies the rising edge as the clock edge.  
FALling specifies the falling edge as the clock edge.

## **BUS:B<x>:PARAllel:CLOCK:ISCLOCKed**

Sets or returns the parallel bus clock function for bus <x>, where x is the bus number.

**Group**    Bus

**Syntax**    `BUS:B<x>:PARAllel:CLOCK:ISCLOCKed {YES|NO}`  
`BUS:B<x>:PARAllel:CLOCK:ISCLOCKed?`

**Arguments**    YES specifies that the parallel bus is clocked.  
NO specifies that the parallel bus is not clocked.

## **BUS:B<x>:PARAllel:CLOCK:SOURce**

Sets or returns the parallel bus source <x>, where x is the bus number.



|                  |   |
|------------------|---|
| <b>Group</b>     | Bus   |
| <b>Syntax</b>    | BUS:B<x>:PARAllel:CLOCK:SOURce {CH1 CH2 CH3 CH4 D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15}<br>BUS:B<x>:PARAllel:CLOCK:SOURce?   |
| <b>Arguments</b> | CH1–CH4 specifies the channel to use as the parallel bit source.<br>D0–D15 specifies the digital channel to use as the parallel bit source. |

## BUS:B<x>:PARAllel:WIDTh

Sets or returns the number of bits used for the width of the parallel bus <x>, where x is the bus number.

|                  |   |
|------------------|---|
| <b>Group</b>     | Bus   |
| <b>Syntax</b>    | BUS:B<x>:PARAllel:WIDTh <NR1><br>BUS:B<x>:PARAllel:WIDTh? |
| <b>Arguments</b> | <NR1> is the number of bits.                              |

## BUS:B<x>:POSition

Sets or returns the position of the bus <x> waveform on the display, where x is the bus number 1 through 4.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO, DPO2EMBD, or DPO2COMP application module. |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | BUS:B<x>:POSition <NR3><br>BUS:B<x>:POSition?                               |
| <b>Arguments</b>  | <NR3> specifies the position.   |

## **BUS:B<x>:RS232C:BITRate**

Sets or returns the RS232 bit rate for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Bus

**Syntax** `BUS:B<x>:RS232C:BITRate <NR1>`  
`BUS:B<x>:RS232C:BITRate?`

**Arguments** <NR1> is the bit rate in bits-per-second: 50 to 10000000. You can enter any positive integer, and the instrument will coerce the value to the closest supported bit rate.

## **BUS:B<x>:RS232C:DATABits**

Sets or returns the number of RS232 data bits for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Bus

**Syntax** `BUS:B<x>:RS232C:DATABits {7|8|9}`  
`BUS:B<x>:RS232C:DATABits?`

**Arguments** 7 specifies seven bits in the RS232 data frame.  
8 specifies eight bits in the RS232 data frame.  
9 specifies nine bits in the RS232 data frame.

## **BUS:B<x>:RS232C:DELIMiter**

Sets or returns the RS232 delimiting value for a packet on bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

|                  |  |
|------------------|--|
| <b>Group</b>     | Bus  |
| <b>Syntax</b>    | BUS:B<x>:RS232C:DELIMiter {NUL LF CR Space XFF}<br>BUS:B<x>:RS232C:DELIMiter?          |
| <b>Arguments</b> | NUL specifies 0x00.<br>LF specifies 0x0A.<br>CR specifies 0x0D.<br>XFF specifies 0xFF. |

## BUS:B<x>:RS232C:DISplaymode

Sets or returns the display mode for the bus <x> display and event table, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2COMP application module.   |
| <b>Group</b>      | Bus  |
| <b>Syntax</b>     | BUS:B<x>:RS232C:DISplaymode {FRAME PACKET}<br>BUS:B<x>:RS232C:DISplaymode?   |
| <b>Arguments</b>  | FRAME displays each frame as a single entity.<br>PACKET displays a group of frames terminated with a single frame defined by the BUS:B<x>:RS232C:DELIMiter command or the front panel. |

## BUS:B<x>:RS232C:PARity

Sets or returns the RS232C parity for bus <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2COMP application module.              |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | BUS:B<x>:RS232C:PARity {NONE EVEN ODD}<br>BUS:B<x>:RS232C:PARity? |

**Arguments** NONE specifies no parity.  
EVEN specifies even parity.  
ODD specifies odd parity.

---

**NOTE.** 9 data bits (BUS:B<x>:DATABITS 9) does not support parity. Changing the parity with the PI while using 9 data bits, has no effect.

---

## BUS:B<x>:RS232C:POLarity

Sets or returns the RS232C polarity for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Bus

**Syntax** BUS:B<x>:RS232C:POLarity {NORMAl|INVERTed}  
BUS:B<x>:RS232C:POLarity?

**Arguments** NORMAl sets the RS232C bus polarity to positive.  
INVERTed sets the RS232C bus polarity to negative.

## BUS:B<x>:RS232C:RX:SOURce

Sets or returns the RS232 RX source for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Bus

**Syntax** BUS:B<x>:RS232C:RX:SOURce {CH1|CH2|CH3|CH4|  
D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}  
BUS:B<x>:RS232C:RX:SOURce?

**Arguments** CH1–CH4 specifies the channel to use for the RS232 RX source.  
D0–D15 specifies the digital channel to use for the RS232 RX source.

**BUS:B<x>:RS232C:TX:SOURce**

Sets or returns the RS232 TX Source for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Bus

**Syntax** `BUS:B<x>:RS232C:TX:SOURce {CH1|CH2|CH3|CH4|  
D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:RS232C:TX:SOURce?`

**Arguments** CH1–CH4 specifies the channel to use as the RS232 TX source.  
D0–D15 specifies the digital channel to use as the RS232 TX source.

**BUS:B<x>:SPI{:CLOCK|:SCLK}:POLARity**

Sets or returns the SPI SCLK polarity for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:SPI{:CLOCK|:SCLK}:POLARity {FALL|RISe}`  
`BUS:B<x>:SPI{:CLOCK|:SCLK}:POLARity?`

**Arguments** FALL specifies the falling edge.  
RISe specifies the rising edge.

**BUS:B<x>:SPI{:CLOCK|:SCLK}:SOURce**

Sets or returns the SPI SCLK source for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:SPI{ :CLOCK| :SCLK}:SOURCE {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:SPI{ :CLOCK| :SCLK}:SOURCE?`

**Arguments** CH1-CH4 is the channel to use as the SPI SCLK source.  
D0-D15 is the digital channel to use as the SPI SCLK source.

## **BUS:B<x>:SPI:DATA{:IN|:MISO}:POLARity**

Sets or returns the SPI MISO polarity for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:SPI:DATA{:IN|:MISO}:POLARity {LOW|HIGH}`  
`BUS:B<x>:SPI:DATA{:IN|:MISO}:POLARity?`

**Arguments** LOW specifies an active low polarity.  
HIGH specifies an active high polarity.

## **BUS:B<x>:SPI:DATA{:IN|:MISO}:SOURCE**

Sets or returns the SPI MISO source for bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:SPI:DATA{:IN|:MISO}:SOURCE {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:SPI:DATA{:IN|:MISO}:SOURCE?`

**Arguments** CH1-CH4 is the channel to use as the SPI MISO source.  
D0-D15 is the digital channel to use as the SPI MISO source.

**BUS:B<x>:SPI:DATA{:OUT|:MOSI}:POLARity**

Sets or returns the SPI MOSI polarity for bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.   |
| <b>Group</b>      | Bus  |
| <b>Syntax</b>     | BUS:B<x>:SPI:DATA{:OUT :MOSI}:POLARity {LOW HIGH}<br>BUS:B<x>:SPI:DATA{:OUT :MOSI}:POLARity? |
| <b>Arguments</b>  | LOW specifies the active low polarity.<br>HIGH specifies the active high polarity.           |

**BUS:B<x>:SPI:DATA{:OUT|:MOSI}:SOURce**

Sets or returns the SPI MOSI source for bus <x>, where x is the bus number>.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | BUS:B<x>:SPI:DATA{:OUT :MOSI}:SOURce {CH1 CH2 CH3 CH4 <br>D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15}<br>BUS:B<x>:SPI:DATA{:OUT :MOSI}:SOURce? |
| <b>Arguments</b>  | CH1-CH4 is the channel to use as the SPI MISO source.<br>D0-D15 is the digital channel to use as the SPI MISO source.                                     |

**BUS:B<x>:SPI{:SElect|:SS}:POLARity**

Sets or returns the SPI SS polarity for bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module. |
| <b>Group</b>      | Bus  |

**Syntax**     `BUS:B<x>:SPI{:SElect|:SS}:POLARity {LOW|HIGH}`  
`BUS:B<x>:SPI{:SElect|:SS}:POLARity?`

**Arguments**     LOW specifies an active low polarity.  
HIGH specifies an active high polarity.

## **BUS:B<x>:SPI{:SElect|:SS}:SOURCE**

Sets or returns the SPI SS source for bus <x>, where x is the bus number.

**Conditions**     This command requires a DPO2EMBD application module.

**Group**     Bus

**Syntax**     `BUS:B<x>:SPI{:SElect|:SS}:SOURCE {CH1|CH2|CH3|CH4|`  
`D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`BUS:B<x>:SPI{:SElect|:SS}:SOURCE?`

**Arguments**     CH1–CH4 is the channel to use as the SPI SS source.  
D0–D15 is the digital channel to use as the SPI SS source.

## **BUS:B<x>:SPI:BITOrder**

Sets or returns the bit order for the specified SPI bus.

**Conditions**     This command requires a DPO2EMBD application module.

**Group**     Bus

**Syntax**     `BUS:B<x>:SPI:BITOrder {LSB|MSB}`  
`BUS:B<x>:SPI:BITOrder?`

**Arguments**     LSB specifies least significant bit first  
MSB specifies most significant bit first



**Examples** `BUS:B1:SPI:BITOrder LSB` sets the bit order to LSB for the specified bus.  
`BUS:B1:SPI:BITOrder?` might return `BUS:B1:SPI:BITOrder MSB` indicating that the returned bit order for the specified bus is MSB.

## BUS:B<x>:SPI:DATA:SIZE

Sets or returns the number of bits per word for the specified SPI bus.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:SPI:DATA:SIZE <NR1>`  
`BUS:B<x>:SPI:DATA:SIZE?`

**Arguments** `<NR1>` specifies the number of bits per word.

**Examples** `BUS:B1:SPI:DATA:SIZE 16` sets the data size to 16.  
`BUS:B1:SPI:DATA:SIZE?` might return `BUS:B1:SPI:DATA:SIZE 8` indicating that the data size is 8 bits.

## BUS:B<x>:SPI:FRAMING

Sets or returns the type of SPI framing.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Bus

**Syntax** `BUS:B<x>:SPI:FRAMING {SS|IDLEtime}`  
`BUS:B<x>:SPI:FRAMING?`

**Arguments** `SS` specifies framing by SS (non 2-wire).  
`IDLEtime` specifies framing by Idle Time (2-wire).

**Examples**    `BUS:B1:SPI:FRAMING SS` sets the SPI framing type to SS.  
`BUS:B1:SPI:FRAMING IDLETIME` might return `BUS:B1:SPI:FRAMING IDLETIME` indicating that the SPI framing type is set to IDLETIME.

## **BUS:B<x>:SPI:IDLETime**

Sets or returns the SPI bus idle time in seconds for the specified SPI bus.

**Conditions**    This command requires a DPO2EMBD application module.

**Group**        Bus

**Syntax**       `BUS:B<x>:SPI:IDLETime <NR3>`  
`BUS:B<x>:SPI:IDLETime?`

**Arguments**    <NR3> specifies the idle time in seconds.

**Examples**     `BUS:B1:SPI:IDLETime?` might return `BUS:B1:SPI:IDLETime 5.0000E-6` indicating that the idle time is 5  $\mu$ s.

## **BUS:B<x>:STATE**

Sets or returns the on/off state of bus <x>, where x is the bus number.

**Group**        Bus

**Syntax**       `BUS:B<x>:STATE {<NR1>|OFF|ON}`  
`BUS:B<x>:STATE?`

**Related Commands**    [SElect:BUS<x>](#)

**Arguments**    ON or <NR1>  $\neq$  0 turns on the bus.  
OFF or <NR1> = 0 turns off the bus.

**BUS:B<x>:TYPE**

Sets or returns the bus type for <x>, where x is the bus number.

|                  |   |
|------------------|---|
| <b>Group</b>     | Bus   |
| <b>Syntax</b>    | BUS:B<x>:TYPE {I2C SPI CAN RS232C PARAllel LIN}<br>BUS:B<x>:TYPE  |
| <b>Arguments</b> | I2C specifies the Inter-IC bus.<br><br>SPI specifies the Serial Peripheral Interface bus (not available on two-channel models).<br><br>CAN specifies the Controller Area Network bus.<br><br>RS232C specifies the RS232C bus.<br><br>PARAllel specifies the parallel bus.<br><br>LIN specifies the LIN bus. |

**BUS:LOWerthreshold:CH<x>**

Sets the lower threshold for each channel. This applies to all search and trigger types that use the channel. This command supersedes the :BUS:THRESHOLD:CH<x> above.

|                  |   |
|------------------|---|
| <b>Group</b>     | Bus   |
| <b>Syntax</b>    | BUS:LOWerthreshold:CH<x> {<NR3> ECL TTL}<br>BUS:LOWerthreshold:CH<x>?   |
| <b>Arguments</b> | <NR3> specifies the threshold in volts.<br><br>ECL specifies a preset ECL high level of -1.3 V.<br><br>TTL specifies a preset TTL high level of 1.4 V.  |
| <b>Examples</b>  | BUS:LOWERTHRESHOLD:CH1 TTL sets the CH1 lower threshold to 800 mV.<br><br>BUS:LOWERTHRESHOLD:CH1? might return :BUS:LOWERTHRESHOLD:CH1 -800.0000E-3 indicating that the CH1 lower threshold is -800 mV. |

## BUS:THReshold:CH<x>

Sets or returns the threshold for analog channel <x>, where x is the channel number. This setting applies to all trigger types that use the channel. The threshold value of CH<x> is set to the closest value used by hardware. The threshold settings for both BUS: THReshold:CH<x> and BUS:THRESHOLD:D<x> channels will be rounded to the nearest threshold value used by hardware and not necessarily the value set by the user. For example: If you set the threshold to 2.48 volts and the hardware only supports 2.5 volts, then the threshold value will be 2.5 volts and not 2.48 as set by you.

**Conditions** This command requires a DPO2AUTO, DPO2EMBD, or DPO2COMP application module.

**Group** Bus

**Syntax** BUS:THReshold:CH<x> {ECL|TTL|<NR3>}  
 BUS:THReshold:CH<x>?

**Related Commands** [TRIGger:A:LEVel:CH<x>](#)

**Arguments** ECL specifies a preset ECL high level of -1.3 V.  
 TTL specifies a TTL preset high level of 1.4 V.  
 <NR3> specifies the threshold level in volts.

## BUS:THReshold:D<x>

Sets or returns the threshold for digital channel <x>, where x is the digital channel number. This will apply to all search and trigger types that use the channel. The threshold value of D<x> is set to the closest value used by hardware.

Digital channel thresholds are set in two groups. One for channels D0–D7 and the other for D8–D15. Setting the thresholds in one of these groups sets the threshold for all the channels in that group.

**Conditions** This command requires a DPO2AUTO, DPO2EMBD, or DPO2COMP application module.

**Group** Bus

**Syntax** `BUS:THRESHOLD:D<x> {<NR3>|ECL|TTL}`  
`BUS:THRESHOLD:D<x>?`

**Related Commands** [TRIGGER:A:LEVEL:D<x>](#)

**Arguments** ECL specifies a preset ECL high level of -1.3 V.  
TTL specifies a preset TTL high level of 1.4 V.  
<NR3> specifies the threshold level in volts.

## BUS:UPPERthreshold:CH<x>

Sets the upper threshold for each channel. This applies to all search and trigger types that use the channel.

**Group** Bus

**Syntax** `BUS:UPPERthreshold:CH<x> {<NR3>|ECL|TTL}`  
`BUS:UPPERthreshold:CH<x>?`

**Arguments** <NR3> specifies the threshold in volts.  
ECL specifies a preset ECL high level of -1.3 V.  
TTL specifies a preset TTL high level of 1.4 V.

**Examples** `BUS:UPPERTHRESHOLD:CH1 800.0000E-3` sets the CH1 upper threshold to 800 mV.  
`BUS:UPPERTHRESHOLD:CH1?` might return `:BUS:UPPERTHRESHOLD:CH1 -800.0000E-3` indicating that the CH1 upper threshold is set to -800 mV.

## BUSY? (Query Only)

Returns the status of the oscilloscope. This command allows you to synchronize the operation of the oscilloscope with your application program.

**Group** Status and Error

**Syntax** `BUSY?`

**Related Commands**    \*OPC, \*WAI

**Returns**    <NR1> = 0 means the oscilloscope is not busy processing a command whose execution time is extensive.  
 <NR1> = 1 means the oscilloscope is busy processing one of the commands listed in the table below.

**Commands that affect BUSY? response**

| Operation                   | Command  |
|-----------------------------|--|
| Single sequence acquisition | ACQUIRE:STATE ON or<br>ACQUIRE:STATE RUN or<br>ACQUIRE:STATE1<br>(when ACQUIRE:STOPAfter is set to SEQUENCE) |
| Hard copy operation         | HARDCOPY START   |
| Calibration step            | Refer to the optional oscilloscope Service Manual.   |

**Examples**    BUSY? might return :BUSY 1 indicating that the oscilloscope is currently busy.

**\*CAL? (Query Only)**

Performs an internal self-calibration and returns the oscilloscope calibration status.

---

**NOTE.** *Disconnect or otherwise remove all input signals prior to starting self-calibration. The self-calibration can take several minutes to complete.*

---

*No other commands are executed until calibration is complete.*

---

**Group**    Calibration and Diagnostic

**Syntax**    \*CAL?

**Returns**    <NR1> = 1 indicates the calibration did not complete successfully.  
 <NR1> = 0 indicates the calibration completed without errors.

**Examples**    \*CAL? starts the internal signal path calibration and might return 0 to indicate that the calibration was successful.

## CALibrate:FACTory (No Query Form)

Provides the controls for starting and stopping the factory calibration process. The factory calibration process consists of a series of steps.

|                  |  |
|------------------|--|
| <b>Group</b>     | Calibration and Diagnostic   |
| <b>Syntax</b>    | CALibrate:FACTory {START CONTInue PREVious ABOrt DUMp}   |
| <b>Arguments</b> | <p>START initializes the factory calibration sequence and starts the first calibration step.</p> <p>CONTInue begins the next factory calibration step.</p> <p>PREVious attempts to run the most recent factory calibration step again.</p> <p>ABOrt stops the calibration process.</p> <p>DUMp stops the calibration and prints the calibration constants.</p> |
| <b>Examples</b>  | CALIBRATE:FACTORY START starts the calibration process.  |

## CALibrate:FACTory:STATus? (Query Only)

Returns the factory calibration status value saved in nonvolatile memory.

|                 |  |
|-----------------|--|
| <b>Group</b>    | Calibration and Diagnostic   |
| <b>Syntax</b>   | CALibrate:FACTory:STATus?  |
| <b>Examples</b> | CALIBRATE:FACTORY:STATus? might return CALIBRATE:FACTORY:STATus PASS indicating that factory calibration passed. |

## CALibrate:INTERNa1 (No Query Form)

This command starts a signal path compensation.

|               |                            |
|---------------|----------------------------|
| <b>Group</b>  | Calibration and Diagnostic |
| <b>Syntax</b> | CALibrate:INTERNa1         |

**Arguments** None

**Examples** CALIBRATE:INTERNAL starts a serial path compensation cycle.

## CALibrate:INTERNAL:START (No Query Form)

Starts the internal signal path calibration (SPC) of the oscilloscope. You can use the [CALibrate:INTERNAL:STATUS?](#) query to return the current status of the internal signal path calibration of the oscilloscope.

**Group** Calibration and Diagnostic

**Syntax** CALIBRATE:INTERNAL:START

**Related Commands** [CALibrate:RESults:SPC?](#)

**Examples** CALIBRATE:INTERNAL:START initiates the internal signal path calibration of the oscilloscope.

## CALibrate:INTERNAL:STATUS? (Query Only)

Returns the current status of the oscilloscope internal signal path calibration for the last SPC operation.

**Group** Calibration and Diagnostic

**Syntax** CALIBRATE:INTERNAL:STATUS?

**Related Commands** [\\*CAL?](#)

**Returns** This query will return one of the following:

- INIT indicates the oscilloscope has not had internal signal path calibration run.
- PASS indicates the signal path calibration completed successfully.
- FAIL indicates the signal path calibration did not complete successfully.
- RUNNING indicates the signal path calibration is currently running.



**Examples** CALIBRATE:INTERNAL:STATUS? might return  
:CALIBRATE:INTERNAL:STATUS INIT indicating that the current status of the internal signal path calibration is that it has not been run.

## CALibrate:RESuLts? (Query Only)

Returns the status of internal and factory calibrations, without performing any calibration operations. The results returned do not include the calibration status of attached probes. The CALibrate:RESuLts? query is intended to support GO/NoGO testing of the oscilloscope calibration readiness: all returned results should indicate PASS status if the oscilloscope is "fit for duty". It is quite common, however, to use uncalibrated probes (particularly when the oscilloscope inputs are connected into a test system with coaxial cables).

**Group** Calibration and Diagnostic

**Syntax** CALibrate:RESuLts?

**Related Commands** [\\*CAL?](#)

## CALibrate:RESuLts:FACTory? (Query Only)

Returns the status of internal and factory calibration, without performing any calibration operations.

**Group** Calibration and Diagnostic

**Syntax** CALibrate:RESuLts:FACTory?

## CALibrate:RESuLts:SPC? (Query Only)

Returns the status of the SPC operation. This query does not initiate a SPC.

**Group** Calibration and Diagnostic

**Syntax** CALibrate:RESuLts:SPC?

**Related Commands** [\\*CAL?](#)

**Returns** INIT indicates that SPC has never successfully completed.  
PASS indicates that the last SPC operation passed.  
FAIL indicates that the last SPC operation failed.  
RUNNING indicates that the SPC operation is running.

**Examples** CALIBRATE:RESULTS:SPC? returns the results of the last SPC operation: either PASS or FAIL.

### CALibrate:TEMPerature? (Query Only)

Returns 0 as the DPO/MSO2000 series instruments do not support recording of oscilloscope temperature.

**Group** Calibration and Diagnostic

**Syntax** CALibrate:TEMPerature?

### CH<x>? (Query Only)

Returns the vertical parameters for channel <x>, where x is the channel number.

**Group** Vertical

**Syntax** CH<x>?

### CH<x>:BANdwidth

Sets or returns the selectable low-pass bandwidth limit filter for channel <x>, where x is the channel number.

**Group** Vertical

**Syntax** CH<x>:BANdwidth {TWEnty|FUL1|<NR3>}  
CH<x>:BANdwidth?

**Arguments** TWENTY sets the upper bandwidth limit of channel <x> to 20 MHz.

FULL disables any optional bandwidth limiting. The specified channel operates at its maximum attainable bandwidth.

<NR3> is a double-precision ASCII string. The oscilloscope rounds this value to an available bandwidth using geometric rounding, and then uses this value to set the upper bandwidth limit.

---

**NOTE.** *Other values may be possible depending on the attached probes.*

---

**Examples** CH1:BANDWIDTH TWENTY sets the bandwidth of channel 1 to 20 MHz.

## CH<x>:COUPLing

Sets or returns the input attenuator coupling setting for channel <x>, where x is the channel number.

**Group** Vertical

**Syntax** CH<x>:COUPLing {AC|DC|GND}  
CH<x>:COUPLing?

**Arguments** AC sets channel <x> to AC coupling.

DC sets channel <x> to DC coupling.

GND sets channel<x> to ground. Only a flat, ground-level waveform will be displayed.

**Examples** CH2:COUPLING GND sets channel 2 to ground.

CH3:COUPLing? might return :CH3:COUPLing DC indicating that channel 3 is set to DC coupling.

## CH<x>:DESKew

Sets or returns the deskew time for channel <x>, where x is the channel number. You can adjust the deskew time to add an independent, channel-based delay time to the delay (set by the horizontal position control and common to all channels) from the common trigger point to first sample taken for each channel. This lets you compensate individual channels for different delays introduced by their individual input hook ups.

|                  |  |
|------------------|--|
| <b>Group</b>     | Vertical   |
| <b>Syntax</b>    | CH<x>:DESKew <NR3><br>CH<x>:DESKew?  |
| <b>Arguments</b> | <NR3> is the deskew time for channel <x>, ranging from -100 ns to +100 ns with a resolution of 1 ns.   |
| <b>Examples</b>  | CH4:DESKew 5.0E-9 sets the deskew time for channel 4 to 5 ns.<br><br>CH2:DESKew? might return :CH2:DESKEW 2.0000E-09 indicating that the deskew time for channel 2 is set to 2 ns. |

## CH<x>:IMPedance

Sets or returns the input impedance of channel <x>, where x is the channel number. This command is provided for compatibility.

|               |                                     |
|---------------|-------------------------------------|
| <b>Group</b>  | Vertical                            |
| <b>Syntax</b> | CH<x>:IMPedance<br>CH<x>:IMPedance? |

## CH<x>:INVert

Sets or returns the invert function for channel <x>, where is the channel number. When on, the invert function inverts the waveform for the specified channel.

---

**NOTE.** *This command inverts the waveform for display purposes only. The oscilloscope does not use an inverted waveform for triggers or trigger logic inputs.*

---

|                  |  |
|------------------|--|
| <b>Group</b>     | Vertical   |
| <b>Syntax</b>    | CH<x>:INVert {ON OFF}<br>CH<x>:INVert?   |
| <b>Arguments</b> | OFF sets the invert function for channel <x> to off.<br><br>ON sets the invert function for channel <x> to on. |

**Examples** CH4:INVERT ON inverts the waveform on channel 4.  
 CH2:INVERT? might return :CH2:INVERT 0 indicating that channel 2 is not inverted.

## CH<x>:LABel

Sets or returns the waveform label for channel <x>, where x is the channel number.

**Group** Vertical

**Syntax** CH<x>:LABel <Qstring>  
 CH<x>:LABel?

**Arguments** <Qstring> is an alpha-numeric string of text, enclosed in quotes, that contains the text label information for the channel <x> waveform. The text string is limited to 30 characters.

## CH<x>:OFFSet

Sets or returns the vertical offset for channel <x>, where x is the channel number.

This command offsets the vertical acquisition window (moves the level at the vertical center of the acquisition window) for the specified channel. Visualize offset as scrolling the acquisition window towards the top of a large signal for increased offset values, and scrolling towards the bottom for decreased offset values. The resolution of the vertical window sets the offset increment for this control.

Offset adjusts only the vertical center of the acquisition window for channel waveforms to help determine what data is acquired. The oscilloscope always displays the input signal minus the offset value.

The channel offset range depends on the vertical scale factor. The valid ranges for the DPO/MSO2000 series oscilloscopes are (when the probe and external attenuation factor is X1):

- For V/Div settings from 2 mV/div to 200 mV/div, the offset range is +/- 1 V
- For V/Div settings from 202 mV/div to 5 V/div, the offset range is +/- 25 V

**Group** Vertical

**Syntax** CH<x>:OFFSet <NR3>  
 CH<x>:OFFSet?

**Related Commands** [CH<x>:POSition](#)

**Arguments** <NR3> is the offset value for the specified channel <x>.

**Examples** CH3:OFFSet 2.0E-3 sets the offset for channel 3 to 2 mV.  
 CH4:OFFSet? might return :CH4:OFFSET 1.0000E-03 indicating that the offset for channel 4 is set to 1 mV.

## CH<x>:POSition

Sets or returns the vertical position of channel <x>, where x is the channel number. The position value is applied to the signal before it is digitized.

Increasing the position value of a waveform causes the waveform to move up. Decreasing the position value causes the waveform to move down. The position value determines the vertical graticule coordinate at which input signal values, minus the present offset setting for that channel, are displayed. For example, if the position for Channel 3 is set to 2.0 and the offset is set to 3.0, then input signals equal to 3.0 units are displayed 2.0 divisions above the center of the screen (at 1 V/div).

**Group** Vertical

**Syntax** CH<x>:POSition <NR3>  
 CH<x>:POSition?

**Related Commands** [CH<x>:OFFSet](#), [REF<x>:VERTical:POSition](#), [MATH\[1\]:VERTical:POSition](#)

**Arguments** <NR3> is the position value for channel <x>, in divisions, from the center graticule. The range is 4 to -4 divisions.

**Examples** CH2:POSition 1.3 positions the Channel 2 input signal 1.3 divisions above the center graticule.  
 CH1:POSition? might return :CH1:POSITION -1.3000 indicating that the current position of Channel 1 is 1.3 divisions below the center graticule.

## CH<x>:PRObe? (Query Only)

Returns all information concerning the probe attached to channel <x>, where x is the channel number.

**Group** Vertical

**Syntax** CH<x>:PRObe?

## CH<x>:PRObe:AUTOZero (No Query Form)

Sets the TekVPI probe attached to channel <x> to zero, where x is the channel number

**Group** Vertical

**Syntax** CH<x>:PRObe:AUTOZero EXECute

**Arguments** EXECute auto zeros the probe.

## CH<x>:PRObe:COMMAND (No Query Form)

Sets the state of the probe control specified with the first argument to the state specified with the second argument. The commands and states are unique to the attached probe type. Only certain VPI probes support this command. See the probe documentation for how to set these string arguments.

**Group** Vertical

**Syntax** CH<x>:PRObe:COMMAND <QString>, <QString>

**Arguments** <QString> are quoted strings specifying the probe command and value to set in the probe attached to the specified channel.

**Examples** CH1:PROBE:COMMAND "MODE", "4-4V1MHZ" sets a Tektronix VPI-DPG probe to the 4-4V1MHz mode.

CH1:PROBE:COMMAND "OUTPUT", "ON" turns the output of a Tektronix VPI-DPG probe on.

CH1:PROBE:COMMAND?"MODE" might return CH1:PROBE:COMMAND "MODE", "4-4V1MHZ".

## CH<x>:PRObe:DEGAUss (No Query Form)

Starts a degauss auto-zero cycle on a TekVPI current probe attached to the input channel specified by <x>, where x is the channel number.

**Group** Vertical

**Syntax** CH<x>:PRObe:DEGAUss EXECute

**Arguments** EXECute initiates the degauss operation.

## CH<x>:PRObe:DEGAUss:STATE? (Query Only)

Returns the state of the probe degauss for the channel specified by <x>, where x is the channel number.

---

**NOTE.** This command will return *PASSED* for probes that do not support degauss operations.

---

**Group** Vertical

**Syntax** CH<x>:PRObe:DEGAUss:STATE?

**Returns** NEEDED indicates the probe should be degaussed before taking measurements.  
 RECOMMENDED indicates the measurement accuracy might be improved by degaussing the probe.  
 PASSED indicates the probe is degaussed.  
 FAILED indicates the degauss operation failed.  
 RUNNING indicates the probe degauss operation is currently in progress.

## CH<x>:PRObe:FORCEDRange

Sets or returns the range of a TekVPI probe attached to the channel specified by <x>, where x is the channel number.

---

**NOTE.** This command will return *PASSED* for probes that do not support degauss operations.

---



|                  |   |
|------------------|---|
| <b>Group</b>     | Vertical  |
| <b>Syntax</b>    | CH<x>:PRObe:FORCEDRange <NR3><br>CH<x>:PRObe:FORCEDRange?             |
| <b>Arguments</b> | <NR3> specifies the range, which is probe specific.                   |
| <b>Returns</b>   | This command returns 0.0 for probes that do not support forced range. |

## CH<x>:PRObe:GAIN

Sets or returns the gain factor for the probe attached to the channel specified by <x>, where x is the channel number. The "gain" of a probe is the output divided by the input transfer ratio. For example, a common 10x probe has a gain of 0.1.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Vertical  |
| <b>Syntax</b>           | CH<x>:PRObe:GAIN <NR3><br>CH<x>:PRObe:GAIN?   |
| <b>Related Commands</b> | <a href="#">CH&lt;x&gt;:SCAlE</a>   |
| <b>Arguments</b>        | <NR3> is the probe gain. Allowed values depend on the specific probe.   |
| <b>Examples</b>         | CH2:PROBE:GAIN? might return :CH2:PROBE:GAIN 0.1000E+00 indicating that the attached 10x probe delivers 1 V to the channel 2 BNC for every 10 V applied to the probe input. |

## CH<x>:PRObe:ID? (Query Only)

Returns the type and serial number of the probe attached to channel <x>, where x is the channel number.

|               |                 |
|---------------|-----------------|
| <b>Group</b>  | Vertical        |
| <b>Syntax</b> | CH<x>:PRObe:ID? |

**Examples** CH2:PROBE:ID? might return :CH2:PROBE:ID:TYPE "10X";SERNUMBER "N/A" indicating that a passive 10x probe of unknown serial number is attached to channel 2.

## CH<x>:PRObe:ID:SERnumber? (Query Only)

Returns the serial number of the probe attached to channel <x>, where x is the channel number.

---

**NOTE.** For Level 0 and 1 probes, the serial number will be "".

---

**Group** Vertical

**Syntax** CH<x>:PRObe:ID:SERnumber?

**Examples** CH1:PROBE:ID:SERNUMBER? might return :CH1:PROBE:ID:SERNUMBER "B010289" indicating that the serial number of the probe attached to channel 1 is B010289.

## CH<x>:PRObe:ID:TYPE? (Query Only)

Returns the type of probe attached to the channel specified by <x>, where x is the channel number. Level 2 (or higher) probes supply their exact product nomenclature; for Level 0 or 1 probes, a generic "No Probe Detected" message is returned.

**Group** Vertical

**Syntax** CH<x>:PRObe:ID:TYPE?

**Examples** CH1:PROBE:ID:TYPE? might return :CH1:PROBE:ID:TYPE "P6203" indicating that P6203-type probe is attached to channel 1.

## CH<x>:PRObe:RESistance? (Query Only)

Returns the resistance factor of the probe attached to channel <x>, where x is the channel number.

|                 |  |
|-----------------|--|
| <b>Group</b>    | Vertical   |
| <b>Syntax</b>   | CH<x>:PRObe:RESistance?  |
| <b>Examples</b> | CH2:PRObe:RESistance? might return :CH2:PROBE:RESISTANCE<br>10.0000E+6 indicating that the input resistance of the probe attached to Channel<br>2 is 1 MΩ. |

---

**NOTE.** This query will return 0.0 if no probe is attached or the attached probe does not report the input resistance.

---

## CH<x>:PRObe:SIGnal

Sets or returns the input bypass setting of a TekVPI probe attached to channel <x>, where x is the channel number. The probe must support input bypass, for example TCP0001. This command is ignored if sent to an unsupported probe.

|                  |  |
|------------------|--|
| <b>Group</b>     | Vertical   |
| <b>Syntax</b>    | CH<x>:PRObe:SIGna] {BYPass PASS}<br>CH<x>:PRObe:SIGna]?                    |
| <b>Arguments</b> | BYPASS sets the probe to Bypass mode.<br>PASS sets the probe to Pass mode. |

## CH<x>:PRObe:UNIts? (Query Only)

Returns a string describing the units of measure for the probe attached to channel <x>, where x is the channel number.

|                 |   |
|-----------------|---|
| <b>Group</b>    | Vertical  |
| <b>Syntax</b>   | CH<x>:PRObe:UNIts?  |
| <b>Examples</b> | CH4:PROBE:UNITS? might return :CH4:PROBE:UNITS "V" indicating that the<br>units of measure for the probe attached to channel 4 are volts. |

## CH<x>:SCALE

Sets or returns the vertical scale for the channel specified by <x>, where x is the channel number.

Each waveform has a vertical scale parameter. For a signal with constant amplitude, increasing the Scale causes the waveform to be displayed smaller. Decreasing the scale causes the waveform to be displayed larger.

Scale affects all waveforms, but affects channel waveforms differently from other waveforms:

- For channel waveforms, this setting controls the vertical size of the acquisition window as well as the display scale. The range and resolution of scale values depends on the probe attached and any other external factors you have specified.
- For reference and math waveforms, this setting controls the display only, graphically scaling these waveforms and having no affect on the acquisition hardware.

**Group** Vertical

**Syntax** CH<x>:SCALE <NR3>  
CH<x>:SCALE?

**Related Commands** [CH<x>:OFFSet](#), [CH<x>:POSition](#), [REF<x>:VERTical:SCALE](#), [MATH\[1\]:VERTical:SCALE](#)

**Arguments** <NR3> is the vertical channel scale in units-per-division. The value entered here is truncated to three significant digits.

**Examples** CH4:SCALE 100E-03 sets the channel 4 scale to 100 mV per division.  
CH2:SCALE? might return :CH2:SCALE 1.0000 indicating that the current scale setting of channel 2 is 1 V per division.

## CH<x>:TERmination

Sets the connected-disconnected status of a 50  $\Omega$  resistor, which may be connected between the specified channel's coupled input and oscilloscope ground. The channel is specified by <x>. There is also a corresponding query that requests the termination parameter and translates this enumeration into one of the two float values. This command is maintained for compatibility.

**Group** Vertical

**Syntax** CH<x>:TERmination  
CH<x>:TERmination?

## CH<x>:VOLts

Sets or returns the vertical sensitivity for channel <x>, where x is the channel number.

**Group** Vertical

**Syntax** CH<x>:VOLts <NR3>  
CH<x>:VOLts?

**Arguments** <NR3> is the vertical sensitivity, in volts.

## CH<x>:YUNits

Sets or returns the units for the channel specified by <x>, where x is the channel number. String arguments are case insensitive and any unsupported units will generate an error.

Supported units are:

%, /Hz, A, A/A, A/V, A/W, A/dB, A/s, AA, AW, AdB, As, B, Hz, IRE, S/s, V, V/A, V/V, V/W, V/dB, V/s, VV, VW, VdB, Volts, Vs, W, W/A, W/V, W/W, W/dB, W/s, WA, WV, WW, WdB, Ws, dB, dB/A, dB/V, dB/W, dB/dB, dBA, dBV, dBW, dBdB, day, degrees, div, hr, min, ohms, percent, s

**Group** Vertical

**Syntax** CH<x>:YUNits <QString>  
CH<x>:YUNits?

**Arguments** QString is a string of text surrounded by quotes, specifying the supported units.

## CLEARMenu (No Query Form)

Clears the current menu from the display. This command is equivalent to pressing the front panel Menu off.

**Group** Miscellaneous

**Syntax** CLEARMenu

## \*CLS (No Query Form)

Clears the following:

- Event Queue
- Standard Event Status Register
- Status Byte Register (except the MAV bit)

If the \*CLS command immediately follows an <EOI>, the Output Queue and MAV bit (Status Byte Register bit 4) are also cleared. The MAV bit indicates that information is in the output queue. The device clear (DCL) GPIB control message will clear the output queue and thus MAV. \*CLS does not clear the output queue or MAV.

\*CLS can suppress a Service Request that is to be generated by an \*OPC. This will happen if a single sequence acquisition operation is still being processed when the \*CLS command is executed.

**Group** Status and Error

**Syntax** \*CLS

**Related Commands** [DESE](#), [\\*ESE](#), [\\*ESR?](#), [EVENT?](#), [EVMsg?](#), [\\*SRE](#), [\\*STB?](#)

**Examples** \*CLS clears the oscilloscope status data structures.

## CURSor?

Returns all of the current cursor settings.

**Group** Cursor

**Syntax** CURSor?

**Examples**    `CURSOR?` might return the following as the current cursor settings:  
`:CURSOR:FUNCTION SCREEN;HBARS:POSITION1 0.0000;POSITION2 0.0000;UNITS BASE;:CURSOR:MODE INDEPENDENT;VBARS:POSITION1 -19.0006E-6;POSITION2 -18.9994E-6;UNITS SECONDS`

## CURSOR:FUNCTION

Sets or returns the cursor type. Cursors are attached to the selected waveform in Waveform mode and are attached to the display area in Screen mode.

**Group**    Cursor

**Syntax**    `CURSOR:FUNCTION {OFF|SCREEN|WAVEFORM|VBARS|HBARS}`  
`CURSOR:FUNCTION?`

**Arguments**    `OFF` removes the cursors from the display but does not change the cursor type.

`SCREEN` specifies both horizontal and vertical bar cursors, which measure the selected waveform in horizontal and vertical units. Use these cursors to measure anywhere in the waveform display area.

`WAVEFORM` specifies paired cursors in YT display format for measuring waveform amplitude and time. In XY and XYZ format, these cursors indicate the amplitude positions of an XY pair (Ch1 vs Ch2 voltage, where Ch1 is the X axis and Ch2 is the Y axis) relative to the trigger.

`VBARS` specifies the vertical bar cursor to measure the selected waveform in vertical units.

`HBARS` specifies the horizontal bar cursor to measure the selected waveform in horizontal units.

**Examples**    `CURSOR:FUNCTION WAVEFORM` selects the paired cursors for measuring waveform amplitude and time.

`CURSOR:FUNCTION?` might return `:CURSOR:FUNCTION SCREEN` indicating that the screen cursors are currently selected.

## CURSOR:HBARS? (Query Only)

Returns the current settings for the horizontal bar cursors.

**Group**    Cursor

**Syntax**    `CURSor:HBARs?`

**Examples**    `CURSOR:HBARS?` might return the horizontal bar setting as `:CURSOR:HBARS:POSITION1 320.0000E-03;POSITION2-320.0000E-03;UNITS BASE`

## **CURSor:HBARs:DELTA? (Query Only)**

Returns the vertical difference between the two horizontal bar cursors.

**Group**    Cursor

**Syntax**    `CURSor:HBARs:DELTA?`

**Related Commands**    [CURSor:HBARs:UNIts](#)

**Returns**    A floating point value with an exponent.

**Examples**    `CURSOR:HBARS:DELTA?` might return `:CURSOR:HBARS:DELTA 5.0800E+00` indicating that the difference between the two cursors is 5.08.

## **CURSor:HBARs:POSITION<x>**

Sets or returns the horizontal bar cursor position relative to ground, which is expressed in vertical units (usually volts). The cursor is specified by *x*, which can be 1 or 2.

**Group**    Cursor

**Syntax**    `CURSor:HBARs:POSITION<x> <NR3>`  
`CURSor:HBARs:POSITION<x>?`

**Related Commands**    [CURSor:FUNCTion](#)

**Arguments**    `<NR3>` specifies the cursor position relative to ground.



**Examples** `CURSOR:HBARS:POSITION1 25.0E-3` positions Cursor 1 of the horizontal cursors at 25 mV.

`CURSOR:HBARS:POSITION2?` might return `:CURSOR:HBARS:POSITION2 -64.0000E-03` indicating that Cursor 2 of the horizontal bar cursors is at -64 mV.

## CURSor:HBArS:UNIts

Sets or returns the units for the horizontal bar cursors.

**Group** Cursor

**Syntax** `CURSor:HBArS:UNIts {BASE|PERcent}`  
`CURSor:HBArS:UNIts?`

**Arguments** `BASE` selects the vertical units for the selected waveform.  
`PERcent` selects ratio cursors.

**Examples** `CURSOR:HBARS:UNITS` might return `:CURSOR:HBARS:UNITS BASE` indicating that the units for the horizontal bar cursors are base.

## CURSor:HBArS:USE (No Query Form)

Sets the horizontal bar cursor measurement scale. This command is only applicable when ratio cursors are on.

**Group** Cursor

**Syntax** `CURSor:HBArS:USE {CURrent|HALFgrat|FIVEDivs}`

**Related Commands** [CURSor:HBArS:UNIts](#)

**Arguments** `CURrent` sets the H Bar measurement scale so that 0% is the current position of the lowest H Bar cursor and 100% is the current position of the highest H Bar cursor.

`HALFgrat` resets the H bar measurement scale to half the number of divisions (five for some models and four for others) so that 25% is the current position of the lowest H Bar cursor and 75% is the current position of the highest H Bar.

`FIVEDIVS` sets H Bar measurement scale so that five screen major divisions is 100%, where 0% is -2.5 divisions and 100% is +2.5 divisions from the center horizontal graticule.

**Examples** `CURSOR:HBARS:USE FIVEDIVS` sets the H Bar measurement scale so that 5 screen major divisions equals 100%.

## **CURSOR:MODE**

Sets or returns whether the two cursors move linked together in unison or separately. This applies to the Waveform cursors display mode.

**Conditions** This command is only applicable when waveform cursors are displayed.

**Group** Cursor

**Syntax** `CURSOR:MODE {TRACK|INDEPENDENT}`  
`CURSOR:MODE?`

**Arguments** `TRACK` ties the navigational functionality of the two cursors together. For cursor 1 adjustments, this ties the movement of the two cursors together; however, cursor 2 continues to move independently of cursor 1.

`INDEPENDENT` allows independent adjustment of the two cursors.

**Examples** `CURSOR:MODE TRACK` specifies that the cursor positions move in unison.  
`CURSOR:MODE?` might return `:CURSOR:MODE TRACK` indicating that the two cursors move in unison.

## **CURSOR:VBARS? (Query Only)**

Returns the current settings for the vertical bar cursors.

**Group** Cursor

**Syntax** `CURSOR:VBARS?`

**Examples**    `CURSOR:VBARS?` might return the following vertical bar settings  
`:CURSOR:VBARS:UNITS SECONDS;POSITION1 1.0000E-06;POSITION2`  
`9.0000E-06`

## **CURSor:VBArS:ALTERNATE<x>? (Query Only)**

Returns the alternate readout for the waveform (Vbar) cursors specified by <x>. This alternate readout is in effect for a bus waveform.

**Group**    Cursor

**Syntax**    `CURSor:VBArS:ALTERNATE<x>?`

**Arguments**    `x = 1` specifies vertical bar cursor 1.  
`x = 2` specifies vertical bar cursor 2.

## **CURSor:VBArS:DELTA? (Query Only)**

Returns the horizontal difference between the two vertical bar cursors. The units are specified by the `CURSor:VBArS:UNItS` command.

This is equivalent to watching the cursor readout in the display while using the appropriate cursor mode.

**Group**    Cursor

**Syntax**    `CURSor:VBArS:DELTA?`

**Related Commands**    [CURSor:VBArS:UNItS](#)

**Returns**    <NR3>

**Examples**    `CURSOR:VBARS:DELTA?` might return `:CURSOR:VBARS:DELTA 1.0640E+00` indicating that the time between the vertical bar cursors is 1.064 s.

## **CURSor:VBArS:HPOS<x>? (Query Only)**

Returns the horizontal value of the specified vertical bar ticks for cursor <x>.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Cursor  |
| <b>Syntax</b>           | CURSOR:VBARS:HPOS<x>?   |
| <b>Related Commands</b> | <a href="#">CURSOR:VBARS:UNITS</a>  |
| <b>Arguments</b>        | <x> specifies the cursor. Valid values are 1 and 2.   |
| <b>Returns</b>          | <NR3> indicates the value of one of the ticks. The units are specified by the CURSOR:VBARS:UNITS command.       |
| <b>Examples</b>         | CURSOR:VBARS:HPOS2? might return CURSOR:VBARS:HPOS2 100E-3, indicating that the value of one vertical bar tick. |

## CURSOR:VBARS:POSITION<x>

Sets or returns the horizontal position for the specified vertical bar cursor. The cursor is specified by <x>, which can be 1 or 2. Values are with respect to trigger position or the zero reference point for the designated waveform (if horizontal units are not set to time). Use the CURSOR:VBARS:UNITS command to specify units.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Cursor  |
| <b>Syntax</b>           | CURSOR:VBARS:POSITION<x> <NR3><br>CURSOR:VBARS:POSITION<x>?                       |
| <b>Related Commands</b> | <a href="#">CURSOR:VBARS:UNITS</a>  |
| <b>Arguments</b>        | <NR3> specifies the cursor position.  |
| <b>Returns</b>          | A floating point value with an exponent.  |
| <b>Examples</b>         | CURSOR:VBARS:POSITION2 9.00E-6 positions the cursor2 vertical bar cursor at 9 ms. |

`CURSOR:VBARS:POSITION1?` this command might return  
`:CURSOR:VBARS:POSITION1 1.0000E-06` indicating that the  
 cursor1 vertical bar is positioned at 1  $\mu$ s.

## CURSOR:VBARS:UNITS

Sets or returns the units for the vertical bar cursors.

|                  |   |
|------------------|---|
| <b>Group</b>     | Cursor  |
| <b>Syntax</b>    | <code>CURSOR:VBARS:UNITS {SECONDS HERTZ DEGREES PERCENT}</code><br><code>CURSOR:VBARS:UNITS?</code>   |
| <b>Arguments</b> | <code>SECONDS</code> sets the units of the vertical bar cursors for the time domain (seconds).<br><code>HERTZ</code> sets the units of the vertical bar cursors for the frequency domain (Hertz).<br><code>DEGREES</code> sets the units to degrees for use with an XY display.<br><code>PERCENT</code> sets the units to percent for use with ratio cursors. |
| <b>Returns</b>   | <code>SECONDS</code> , <code>HERTZ</code> , <code>DEGREES</code> , or <code>PERCENT</code> , depending on the current vertical bar cursor units.  |
| <b>Examples</b>  | <code>CURSOR:VBARS:UNITS HERTZ</code> sets the units of the VBARS cursors to 1/seconds.<br><code>CURSOR:VBARS:UNITS?</code> might return <code>:CURSOR:VBARS:UNITS SECONDS</code> indicating that the units for the vertical bar cursor are currently set to seconds.   |

## CURSOR:VBARS:USE (No Query Form)

Sets the vertical bar cursor measurement scale.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command is only applicable when ratio cursors are on. |
| <b>Group</b>      | Cursor   |
| <b>Syntax</b>     | <code>CURSOR:VBARS:USE {CURRENT HALFGRAT FIVEDIVS}</code>  |

**Related Commands**    [CURSOR:VBARS:UNITS](#)

**Arguments**     **CURRENT** sets the V Bar measurement scale so that 0% is the current position of the left-most V Bar cursor and 100% is the current position of the right-most V Bar cursor.

**HALFgrat** resets the V Bar measurement scale to half the number of divisions (five for some models and four for others) so that 25% is the current position of the lowest V Bar cursor and 75% is the current position of the highest V Bar.

**FIVEDivs** sets V Bar measurement scale so that five screen major divisions is 100%, where 0% is -2.5 divisions and 100% is +2.5 divisions from the center vertical graticule.

**Examples**     **CURSOR:VBARS:USE CURRENT** sets the V Bar measurement scale to use the current cursor positions as 0% and 100% of scale if units are set to %.

## **CURSOR:VBARS:VDELTA? (Query Only)**

Returns the vertical difference between the two vertical bar cursor ticks.

**Group**     Cursor

**Syntax**     **CURSOR:VBARS:VDELTA?**

**Related Commands**     [CURSOR:HBAARS:UNITS](#)

**Returns**     <NR3> indicates the horizontal difference between the two vertical bar cursors.

**Examples**     **CURSOR:VBARS:VDELTA?** might return **CURSOR:VBARS:VDELTA 1.064E+0**, indicating that the vertical difference between the vertical bar cursors ticks is 1.064 units.

## **CURSOR:XY:POLAR:RADIUS:DELTA? (Query Only)**

Returns the difference between the cursors X radius and the cursor Y radius ( $\Delta Y$ ,  $\Delta X$ ). The ratio is calculated as  $(\text{cursor 2 Y} - \text{cursor 1 Y}) \div (\text{cursor 2 X} - \text{cursor 1 X})$ .

**Group**     Cursor

**Syntax**     **CURSOR:XY:POLAR:RADIUS:DELTA?**

**CURSOr:XY:POLAr:RADIUS:POSITION<x>? (Query Only)**

Returns the polar radius for the specified cursor, where x can be either 1 or 2.

**Group** Cursor

**Syntax** CURSOr:XY:POLAr:RADIUS:POSITION<x>?

**CURSOr:XY:POLAr:RADIUS:UNItS? (Query Only)**

Returns the polar radius units.

**Group** Cursor

**Syntax** CURSOr:XY:POLAr:RADIUS:UNItS?

**CURSOr:XY:POLAr:THETA:DELta? (Query Only)**

Returns the XY cursor polar angle delta.

**Group** Cursor

**Syntax** CURSOr:XY:POLAr:THETA:DELta?

**CURSOr:XY:POLAr:THETA:POSITION<x>? (Query Only)**

Returns the cursor X or cursor Y polar coordinate, where x is either 1 or 2.

**Group** Cursor

**Syntax** CURSOr:XY:POLAr:THETA:POSITION<x>?

**CURSOr:XY:POLAr:THETA:UNItS? (Query Only)**

Returns the cursor coordinate units.

**Group** Cursor

**Syntax** CURSor:XY:POLAr:THETA:UNITs?

### **CURSor:XY:PRODUCT:DELta? (Query Only)**

Returns the difference between the cursors X position and cursor Y position. The  $\Delta X \times \Delta Y$  value is calculated as  $(X2 - X1) \times (Y2 - Y1)$ .

**Group** Cursor

**Syntax** CURSor:XY:PRODUCT:DELta?

### **CURSor:XY:PRODUCT:POSITION<x>? (Query Only)**

Returns the position of the X or Y cursor used to calculate the  $X \times Y$  cursor measurement, Position 1 =  $(X1 \times Y1)$ ; Position 2 =  $(X2 \times Y2)$ . The cursor is specified by x, which can be 1 or 2.

**Group** Cursor

**Syntax** CURSor:XY:PRODUCT:POSITION<x>?

### **CURSor:XY:PRODUCT:UNITs? (Query Only)**

Returns the XY cursor product units.

**Group** Cursor

**Syntax** CURSor:XY:PRODUCT:UNITs?

### **CURSor:XY:RATIO:DELta? (Query Only)**

Returns the ratio of the difference between the cursors X position and cursor Y position ( $\Delta Y, \Delta X$ ). The ratio is calculated as  $(Y2 - Y1) / (X2 - X1)$ .

**Group** Cursor



**Syntax** CURSor:XY:RATIO:DELta?

### **CURSor:XY:RATIO:POSITION<x>? (Query Only)**

Returns the X (horizontal) or Y (vertical) position for the specified cursor, which can be 1 (X) or 2 (Y). The ratio is calculated as Position 1 = (Y1/X1); Position 2 = (Y2/X2).

**Group** Cursor

**Syntax** CURSor:XY:RATIO:POSITION<x>?

### **CURSor:XY:RATIO:UNIts? (Query Only)**

Returns the cursor X and cursor Y units for the ratio measurement.

**Group** Cursor

**Syntax** CURSor:XY:RATIO:UNIts?

### **CURSor:XY:RECTangular:X:DELta? (Query Only)**

Returns the cursor X delta value in rectangular coordinates.

**Group** Cursor

**Syntax** CURSor:XY:RECTangular:X:DELta?

### **CURSor:XY:RECTangular:X:POSITION<x>**

Sets or returns the X rectangular coordinate for cursor 1 or cursor 2. Cursors are specified by x and can be either 1 or 2.

**Group** Cursor

**Syntax** CURSor:XY:RECTangular:X:POSITION<x> <NR3>  
CURSor:XY:RECTangular:X:POSITION<x>?

**Arguments** <NR3> is the coordinate in volts.

### **CURSor:XY:RECTangular:X:UNIts? (Query Only)**

Returns the cursor X rectangular units.

**Group** Cursor

**Syntax** CURSor:XY:RECTangular:X:UNIts?

### **CURSor:XY:RECTangular:Y:DELta? (Query Only)**

Returns The cursor Y delta value in rectangular coordinates.

**Group** Cursor

**Syntax** CURSor:XY:RECTangular:Y:DELta?

### **CURSor:XY:RECTangular:Y:POSITION<x>**

Sets or returns the Y rectangular coordinate for cursor 1 or cursor 2. The cursor is specified by x.

**Group** Cursor

**Syntax** CURSor:XY:RECTangular:Y:POSITION<x> <NR3>  
CURSor:XY:RECTangular:Y:POSITION<x>?

**Arguments** <NR3> is the coordinate in volts.

### **CURSor:XY:RECTangular:Y:UNIts? (Query Only)**

Returns the cursor Y rectangular units.

**Group** Cursor

**Syntax** CURSor:XY:RECTangular:Y:UNITs?

## CURVe

Transfers waveform data to and from the oscilloscope in binary or ASCII format. Each waveform transferred includes a waveform preamble which contains the data format, scale, and associated information.

For analog waveforms, the CURVe? query transfers data from the oscilloscope. The data source is specified by the [DATA:SOUrce](#) command. The first and last data points are specified by the [DATA:STARt](#) and [DATA:STOP](#) commands.

The oscilloscope returns data from the last acquisition if the source is a channel waveform that is being previewed. The data does not reflect the acquisition preview parameters. The user should always follow acquisition parameter changes with a single sequence OPC command prior to CURVe? to ensure the return data reflects the new acquisition parameters.

The CURVe command transfers waveform data to the oscilloscope. The data is stored in the reference memory location specified by [DATA:DESTination](#), starting with the data point specified by [DATA:STARt](#). Only one waveform can be transferred at a time. The waveform will only be displayed if the reference is displayed.

---

**NOTE.** *Transferring large volumes of data to or from the oscilloscope takes time. ASCII waveform transfer is very inefficient.*

---



---

**NOTE.** *For MSO models, reduced resolution waveform data is not available for the digital channels.*

---

**Group** Waveform Transfer

**Syntax** CURVe {<Block>|<asc curve>|DIGital}  
CURVe?

**Related Commands** [DATA:DESTination](#), [DATA:SOUrce](#), [DATA:STARt](#), [DATA:STOP](#), [WFMinpre?](#), [WFMinpre:BYT\\_Nr](#), [WFMOuTpre?](#), [HEADer](#)

**Arguments** <Block> is the waveform data in binary format. The waveform is formatted as: #<x><yyy><data><newline>, where:

<x> is the number of y bytes. For example, if <yyy>=500, then <x>=3)

<yyy> is the number of bytes to transfer if samples are one or two bytes wide. Use the [WFMinpre:BYT\\_Nr](#) command to set the width for waveforms transferred into the oscilloscope. Use [WFMOutpre:BYT\\_Nr](#) to set the width for waveforms transferred out from the oscilloscope.

<data> is the curve data.

<newline> is a single byte new line character at the end of the data.

<asc curve> is the waveform data in ASCII format. The format for ASCII data is <NR1> [,<NR1>...], where each <NR1> represents a data point.

The following "DIGital" argument is used with MSO models:

**DIGital** Sends the data to the specified reference waveform slot, as specified by the [DATA:DESTINATION](#) command. If the data encoding is a binary format, then depending upon the setting of [HEADER](#), the binary block header is prepended to the output data.

**Examples** CURVE? with ASCII encoding, start and stop of 1 and 10 respectively, and a width set to 1 might return :CURVE 61,62,61,60,60,-59,-59,-58,-58,-59

---

**NOTE.** Curve data is transferred from the oscilloscope asynchronously, depending on the length of the curve record. Such transfers may require several seconds to complete. During this period, the oscilloscope will not respond to the user controls. You can interrupt these asynchronous data transfers by sending a device clear message to the oscilloscope or by interrupting the query with another command or query. In order to verify that curve data has been completely transferred, it is recommended that you follow such queries with an [\\*ESR?](#) query to verify there are no error bits set. You can also check the event queue to determine the cause of the error. If the error was caused by an interrupted query, then the asynchronous data transfer was not complete when the [\\*ESR?](#) query was sent. In such cases, it may be necessary to increase the program's time-out value to ensure that all data is transferred and read.

---

## DATA

Sets or returns the format and location of waveform data transferred with the [CURVe?](#) query or [CURVe](#) command.

**Group** Waveform Transfer

**Syntax** DATA {INIT|SNAP}  
DATA?

|                         |   |
|-------------------------|---|
| <b>Related Commands</b> | <a href="#">CURVe</a> , <a href="#">DATA:START</a> , <a href="#">DATA:STOP</a> , <a href="#">DATA:ENCdg</a> , <a href="#">WFMInpre:NR_Pt</a> , <a href="#">WFMOupre:NR_Pt?</a>  |
| <b>Arguments</b>        | <p>INIT initializes the waveform data parameters to their factory defaults except for <a href="#">DATA:STOP</a>, which is set to the current acquisition record length.</p> <p>SNAP sets <a href="#">DATA:START</a> and <a href="#">DATA:STOP</a> to match the current waveform cursor positions.</p> |
| <b>Examples</b>         | <p>DATA? might return :DATA:DESTINATION REF1:ENCDG RIBINARY;SOURCE CH1;START 1;STOP 500;WIDTH 1</p> <p>DATA INIT initializes the waveform data parameters to their factory defaults.</p>  |

## DATA:RESOLution

Sets or returns whether the [CURVe?](#) query returns full resolution records (acquired data) or reduced resolution records (filtered/displayed data).

|                  |  |
|------------------|--|
| <b>Group</b>     | Waveform Transfer  |
| <b>Syntax</b>    | <pre>DATA:RESOLution {FULL REDUced} DATA:RESOLution?</pre>   |
| <b>Arguments</b> | <p>FULL sets the instrument to return the full undecimated record acquired by the instrument. The full resolution records are not subject to the effects of FilterVu. Full resolution record lengths are 100,000, 125,000, 1,000,000 or 1,250,000 points.</p> <p>REDUCed sets the instrument to return short waveforms that are decimated from the full acquisition record. Reduced records are subject to the low pass filtering effects of FilterVu settings. Reduced record lengths range from 800 to 6250 points, depending on the acquisition settings.</p> |
| <b>Examples</b>  | <p>DATA:RESOLUTION? might return :DATA:RESOLUTION: FULL indicating that the <a href="#">CURVe?</a> query returns full resolution records (acquired data).</p> <p>DATA:RESOLUTIONREDUCED sets the <a href="#">CURVe?</a> returns to reduced resolution records.</p>   |

## DATA:COMPosition

Sets or returns the type of data that the [CURVe?](#) query returns.

|                  |  |
|------------------|--|
| <b>Group</b>     | Waveform Transfer  |
| <b>Syntax</b>    | DATA:COMPosition {COMPOSITE_YT COMPOSITE_ENV SINGULAR_YT}<br>DATA:COMPosition?   |
| <b>Arguments</b> | <p>COMPOSITE_YT is of native width 8-bits and is written as 8-bit data in ISF files and REF waveforms. The data can be queried as 1- or 2-byte data in <a href="#">CURVe?</a> queries by setting the data width to 1 or 2.</p> <p>COMPOSITE_ENV is min/max pair data. Its native width is 8-bits and is written as 8-bit data in ISF files and REF waveforms. The data can be queried as 1- or 2-byte data in <a href="#">CURVe?</a> queries by setting the data width to 1 or 2.</p> <p>SINGULAR_YT is 16-bit data which originates as averaged acquired data or math data. It is written to ISF files and REF waveforms as 16-bit data. The data can be queried as 1- or 2- byte data in <a href="#">CURVe?</a> queries by setting the data width to 1 or 2.</p> |
| <b>Examples</b>  | <p>DATA:COMPOSITION? might return :DATA:COMPOSITION SINGULAR_YT indicating that the data type returned by the <a href="#">CURVe?</a> query is singular waveform.</p> <p>DATA:COMPOSITIONCOMPOSITE_YT sets the return data type of <a href="#">CURVe?</a> query as composite waveform.</p>  |

## DATA:COMPosition:AVAILable? (Query Only)

Lists the waveform data types that are available for return from the instrument under the current instrument settings.

|                 |   |
|-----------------|---|
| <b>Group</b>    | Waveform Transfer   |
| <b>Syntax</b>   | DATA:COMPosition:AVAILable?   |
| <b>Returns</b>  | <p>COMPOSITE_YT for sample data.</p> <p>COMPOSITE_ENV for envelope or peak-detect data.</p> <p>SINGULAR_YT for sample data.</p>     |
| <b>Examples</b> | <p>DATA:COMPOSITION:AVAILABLE? might return<br/>DATA:COMPOSITION:AVAILABLE COMPOSITE_YT indicating that the waveform data type.</p> |

## DATA:DESTINATION

Sets or returns the reference memory location for storing waveform data transferred into the oscilloscope by the [CURVe](#) command.

**Group** Waveform Transfer

**Syntax** DATA:DESTINATION REF<x>  
DATA:DESTINATION?

**Related Commands** [CURVe](#)

**Arguments** REF<x> is the reference location where the waveform will be stored.

**Examples** DATA:DESTINATION? might return :DATA:DESTINATION REF2 indicating that reference 2 is the currently selected reference memory location for incoming waveform data. DATA:DESTINATION REF1 indicates that incoming waveform data be stored in reference 1.

## DATA:ENCdg

Sets or returns the format of outgoing waveform data. This command is equivalent to setting [WFMOupre:ENCdg](#), [WFMOupre:BN\\_Fmt](#), and [WFMIupre:FILTErFreq](#). Setting the DATA:ENCdg value causes the corresponding WFMOupre values to be updated and conversely.

---

**NOTE.** *This command and query does not apply to incoming waveform data.*

---

**Group** Waveform Transfer

**Syntax** DATA:ENCdg  
{ASCIi | FASTest | RIBinary | RPBinary | SRIBinary | SRPbinary}  
DATA:ENCdg?

**Related Commands** [WFMOupre:ENCdg](#), [WFMOupre:BN\\_Fmt](#), [WFMIupre:FILTErFreq](#)

**Arguments** ASCIi specifies the ASCII representation for waveform data points. If ASCII is the value, then :BN\_Fmt and :BYT\_Or are ignored.

**FASTest** specifies that the data be sent in the fastest possible manner consistent with maintaining accuracy and is interpreted with respect to the waveform specified by **DATA:SOURce**.

**RIBinary** specifies signed integer data point representation with the most significant byte transferred first.

When **:BYT\_Nr** is 1, the range is from -128 through 127. When **:BYT\_Nr** is 2, the range is from -32,768 through 32,767. Center screen is 0 (zero). The upper limit is the top of the screen and the lower limit is the bottom of the screen. This is the default argument.

**RPBinary** specifies the positive integer data-point representation, with the most significant byte transferred first.

When **:BYT\_Nr** is 1, the range from 0 through 255. When **:BYT\_Nr** is 2, the range is from 0 to 65,535. The center of the screen is 127. The upper limit is the top of the screen and the lower limit is the bottom of the screen.

**SRIbinary** is the same as **RIBinary** except that the byte order is swapped, meaning that the least significant byte is transferred first. This format is useful when transferring data to IBM compatible PCs.

**SRPbinary** is the same as **RPBinary** except that the byte order is swapped, meaning that the least significant byte is transferred first. This format is useful when transferring data to PCs.

**Table 2-36: DATA and WFMOutpre Parameter Settings**

| <b>DATA:ENCdg Setting</b> | <b>WFMOutpre Settings</b> |                |                |                |
|---------------------------|---------------------------|----------------|----------------|----------------|
|                           | <b>:ENCdg</b>             | <b>:BN_Fmt</b> | <b>:BYT_Or</b> | <b>:BYT_NR</b> |
| ASCIi                     | ASC                       | N/A            | N/A            | 1,2            |
| FASTest                   | BIN                       | RI             | MSB            | 1,2            |
| RIBinary                  | BIN                       | RI             | MSB            | 1,2            |
| RPBinary                  | BIN                       | RP             | MSB            | 1,2            |
| SRIbinary                 | BIN                       | RI             | LSB            | 1,2            |
| SRPbinary                 | BIN                       | RP             | LSB            | 1,2            |

**Examples** **DATA:ENCDG?** might return **:DATA:ENCDG SRPBINARY** for the format of the outgoing waveform data.

**DATA:ENCDG RPBinary** sets the data encoding format to be a positive integer where the most significant byte is transferred first.

## DATA:SOURce

Sets or returns the location of the waveform data transferred from the oscilloscope by the **CURVe?** query.



**Group** Waveform Transfer

**Syntax** DATA:SOURCE {CH1|CH2|CH3|CH4|MATH|REF1|REF2|  
D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}  
DATA:SOURCE?

**Related Commands** [CURVe](#)

**Arguments** CH1–CH4 specifies which analog channel data will be transferred from the oscilloscope to the controller, channels 1 through 4.

MATH specifies that the Math waveform data will be transferred from the oscilloscope to the controller.

REF1–REF2 specifies which Reference waveform data will be transferred from the oscilloscope to the controller, waveforms, 1 through 4.

D0–D15 specifies that the returned waveform data is packaged as 4 bytes per sample, representing the states of all digital channels (D0–D15) and the logic states of the analog channels (Ch1–Ch4). Additionally, when MagniVu is on, the state of the trigger is also included. When the :DATA:SOURCE is set to DIGITAL, the :DATA:WIDTH is automatically set to 4.

The format of the returned 4- byte (32 bit) data is as follows:

| BIT      | Description                         |
|----------|-------------------------------------|
| ---      | -----                               |
| 31 (MSB) | - Unused                            |
| 30       | - Unused                            |
| 29       | - Unused                            |
| 28       | - Unused                            |
| 27       | - Unused                            |
| 26       | - Unused                            |
| 25       | - Unused                            |
| 24       | - Unused                            |
| 23       | - Unused                            |
| 22       | - Unused                            |
| 21       | - Unused                            |
| 20       | - Trigger (only when MagniVu is ON) |
| 19       | - Ch4 logic state                   |
| 18       | - Ch3 logic state                   |
| 17       | - Ch2 logic state                   |
| 16       | - Ch1 logic state                   |
| 15       | - D15 state                         |
| 14       | - D14 state                         |
| 13       | - D13 state                         |
| 12       | - D12 state                         |
| 11       | - D11 state                         |
| 10       | - D10 state                         |
| 09       | - D09 state                         |
| 08       | - D08 state                         |
| 07       | - D07 state                         |
| 06       | - D06 state                         |
| 05       | - D05 state                         |
| 04       | - D04 state                         |
| 03       | - D03 state                         |
| 02       | - D02 state                         |
| 01       | - D01 state                         |
| 00 (LSB) | - D00 state                         |

**Examples** DATA:SOURCE? might return :DATA:SOURCE REF2 indicating that the source for the waveform data which is transferred using a CURVe? query is reference 2.

DATA:SOURCE CH1 specifies that the CH1 waveform will be transferred in the next CURVe? query.

## DATA:START

Sets or returns the starting data point for incoming or outgoing waveform transfer. This command allows for the transfer of partial waveforms to and from the oscilloscope.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Waveform Transfer  |
| <b>Syntax</b>           | DATA:START <NR1><br>DATA:START?  |
| <b>Related Commands</b> | <a href="#">CURVe</a> , <a href="#">DATA</a> , <a href="#">DATA:STOP</a> , <a href="#">WFMinpre:NR_Pt</a> , <a href="#">WFMOuppre:NR_Pt?</a>   |
| <b>Arguments</b>        | <p>&lt;NR1&gt; is the first data point that will be transferred, which ranges from 1 to the record length. Data will be transferred from &lt;NR1&gt; to DATA:STOP or the record length, whichever is less. If &lt;NR1&gt; is greater than the record length, the last data point in the record is transferred.</p> <p>DATA:START and DATA:STOP are order independent. When DATA:STOP is greater than DATA:START, the values will be swapped internally for the CURVE? query.</p> |
| <b>Examples</b>         | <p>DATA:START? might return :DATA:START 214 indicating that data point 214 is the first waveform data point that will be transferred.</p> <p>DATA:START 10 specifies that the waveform transfer will begin with data point 10.</p>   |

## DATA:STOP

Sets or returns the last data point that will be transferred when using the [CURVe?](#) query. This command allows for the transfer of partial waveforms from the oscilloscope.

Changes to the record length value are not automatically reflected in the DATA:STOP value. As record length is varied, the DATA:STOP value must be explicitly changed to ensure the entire record is transmitted. In other words, curve results will not automatically and correctly reflect increases in record length if the distance from [DATA:START](#) to DATA:STOP stays smaller than the increased record length.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Waveform Transfer   |
| <b>Syntax</b>           | DATA:STOP <NR1><br>DATA:STOP?   |
| <b>Related Commands</b> | <a href="#">CURVe</a> , <a href="#">DATA</a> , <a href="#">DATA:START</a> , <a href="#">WFMinpre:NR_Pt</a> , <a href="#">WFMOuppre:NR_Pt?</a> |

**Arguments** <NR1> is the last data point that will be transferred, which ranges from 1 to the record length. If <NR1> is greater than the record length, then data will be transferred up to the record length. If both DATA:START and DATA:STOP are greater than the record length, the last data point in the record is returned.

DATA:START and DATA:STOP are order independent. When DATA:STOP is less than DATA:START, the values will be swapped internally for the CURVE? query.

If you always want to transfer complete waveforms, set DATA:START to 1 and DATA:STOP to the maximum record length, or larger.

**Examples** DATA:STOP? might return :DATA:STOP 14900 indicating that 14900 is the last waveform data point that will be transferred.

DATA:STOP 15000 specifies that the waveform transfer will stop at data point 15000.

## DATA:WIDTH

Sets or returns the number of bytes per data point in the waveform transferred using the CURVE command.

**Group** Waveform Transfer

**Syntax** DATA:WIDTH <NR1>  
DATA:WIDTH?

**Related Commands** [CURVE](#)

**Arguments** <NR1> = 1 specifies that there is 1 byte (8 bits) per point.

<NR1> = 2 specifies that there are 2 bytes (16 bits) per point. This format is useful for AVERAGE waveforms.

**Examples** DATA:WIDTH 1 sets the data width to 1 byte per data point for CURVE data.

## DATE

Sets or returns the date the oscilloscope displays.

**Group** Miscellaneous

**Syntax** DATE <QString>  
DATE?

**Related Commands** [TIME](#)

**Arguments** <QString> is a date in the form "yyyy-mm-dd" where yyyy refers to a four-digit year number, mm refers to a two-digit month number from 01 to 12, and dd refers to a two-digit day number in the month.

**Examples** DATE "2006-01-24" specifies that the date is set to January 24, 2006.  
DATE? might return :DATE 2006-01-24 indicating that the current date is set to January 24, 2006.

## \*DDT

Allows you to specify a command or a list of commands that execute when the oscilloscope receives a [\\*TRG](#) command or the GET IEEE488.2 interface message. Define Device Trigger (\*DDT) is a special alias that the \*TRG command uses.

**Group** Miscellaneous

**Syntax** \*DDT {<Block>|<QString>}  
\*DDT?

**Related Commands** [ALIAS](#), [\\*TRG](#)

**Arguments** <Block> is a complete sequence of program messages. The messages can contain only valid commands that must be separated by semicolons and must follow all rules for concatenating commands. The sequence must be less than or equal to 80 characters. The format of this argument is always returned as a query.

<QString> is a complete sequence of program messages. The messages can contain only valid commands that must be separated by semicolons and must follow all rules for concatenating commands. The sequence must be less than or equal to 80 characters.

**Examples** \*DDT #OACQUIRE:STATE RUN specifies that the acquisition system will be started each time a \*TRG command is sent.

## DESE

Sets or returns the bits in the Device Event Status Enable Register (DESER). The DESER is the mask that determines whether events are reported to the Standard Event Status Register (SESR), and entered into the Event Queue. For a detailed discussion of the use of these registers, see Registers.

**Group** Status and Error

**Syntax** DESE <NR1>  
DESE?

**Related Commands** \*CLS, \*ESE, \*ESR?, EVENT?, EVMsg?, \*SRE, \*STB?

**Arguments** <NR1> sets the binary bits of the DESER according to this value, which ranges from 1 through 255. For example, DESE 209 sets the DESER to the binary value 11010001 (that is, the most significant bit in the register is set to 1, the next most significant bit to 1, the next bit to 0, etc.).

The power-on default for DESER is all bits set if \*PSC is 1. If \*PSC is 0, the DESER maintains the previous power cycle value through the current power cycle.

---

**NOTE.** Setting the DESER and ESER to the same value allows only those codes to be entered into the Event Queue and summarized on the ESB bit (bit 5) of the Status Byte Register. Use the \*ESE command to set the ESER.

---

**Examples** DESE 209 sets the DESER to binary 11010001, which enables the PON, URQ, EXE and OPC bits.

DESE? might return :DESE 186, showing that the DESER contains the binary value 10111010.

## DIAG:LOOP:OPTion

Sets the self-test loop option.

**Group** Calibration and Diagnostic

**Syntax** DIAG:LOOP:OPTion {ALWAYS|FAIL|ONFAIL|ONCE|NTIMES}

- Arguments** ALWAYS continues looping until the self tests (diagnostics) are stopped via the front panel or by an oscilloscope command.
- FAIL causes looping until the first self test (diagnostic) failure or until self tests (diagnostics) are stopped.
- ONFAIL causes looping on a specific test group as long as a FAIL status is returned from the test.
- ONCE executes self test (diagnostics test) sequence once.
- NTIMES runs “n” number of loops.

**Examples** `DIAG:LOOP:OPTION ONCE` runs one loop of self tests.

## DIAG:LOOP:OPTION:NTIMES

Sets the self-test loop option to run N times.

**Group** Calibration and Diagnostic

**Syntax** `DIAG:LOOP:OPTION:NTIMES <NR1>`  
`DIAG:LOOP:OPTION:NTIMES?`

**Arguments** <NR1> is the number of self-test loops.

**Examples** `DIAG:LOOP:OPTION:NTIMES 3` sets the self-test loop to run three times.

`DIAG:LOOP:OPTION:NTIMES?` might return `:DIAG:LOOP:OPTION:NTIMES 5`, indicating that the self-test loop is set to run five times.

## DIAG:LOOP:STOP (No Query Form)

Stops the self-test at the end of the current loop.

**Group** Calibration and Diagnostic

**Syntax** `DIAG:LOOP:STOP`

**Examples** `DIAG:LOOP:STOP` stops the self test at the end of the current loop.

## DIAG:RESULT:FLAG? (Query Only)

Returns the pass/fail status from the last self-test sequence execution. Use this query to determine which test(s) has failed.

**Group** Calibration and Diagnostic

**Syntax** DIAG:RESULT:FLAG?

**Related Commands** [DIAG:RESULT:LOG?](#)

**Returns** PASS indicates that all of the selected self (diagnostic) tests have passed.  
 FAIL indicates that at least one of the selected self (diagnostic) tests has failed.

**Examples** DIAG:RESULT:FLAG? returns either DIAG:RESULT:FLAG PASS or FAIL.

## DIAG:RESULT:LOG? (Query Only)

Returns the internal results log from the last self-test sequence execution. The list contains all modules and module interfaces that were tested along with the pass/fail status of each.

**Group** Calibration and Diagnostic

**Syntax** DIAG:RESULT:LOG?

**Related Commands** [DIAG:RESULT:FLAG?](#)

**Returns** <QString> in the following format:  
 <Status>--<Module name>[, <Status>--<Module name>...]

**Examples** DIAG:RESULT:LOG? might return :DIAG:RESULT:LOG "NOT RUN--CPU,NOT RUN--DISPLAY,NOT RUN--FPANEL,NOT RUN--IO,NOT RUN--ACQ,NOT RUN--ROM,NOT RUN--APPKEY"



## DIAg:SElect (No Query Form)

Sets the type of diagnostics grouping.

**Group** Calibration and Diagnostic

**Syntax** DIAg:SElect {ALL|APPKey|CPU|DISpLay|FPAnel|IO|ROM|ACQ}

**Arguments** ALL runs all diagnostic groups.  
APPKey runs just the application key diagnostic group.  
CPU runs just the CPU diagnostic group.  
DISpLay runs just the display circuit diagnostic group.  
FPAnel runs just the front panel diagnostic group.  
IO runs just the IO board diagnostic group.  
ROM runs just the IO board diagnostic group.  
ACQ runs just the acquisition system diagnostic group.

## DIAg:SElect:<function> (No Query Form)

Runs self-tests on the specified system subsystem.

**Group** Calibration and Diagnostic

**Syntax** DIAg:SElect:<function>

**Arguments** <function> specifies a single oscilloscope subsystem on which to run self tests (diagnostics). Valid values are:  
ACQ tests the acquisition system.  
APPKey tests the application keys.  
CPU tests the CPU.  
DISpLay tests the display.  
FPAnel tests the front panel controls.  
IO tests the IO ports.  
ROM tests the system read only memory.

**Examples**    `DIAG:SELECT:CPU` sets the oscilloscope to run just CPU tests.

## DIAG:STATE (No Query Form)

This command starts or stops the oscilloscope self-test. Depending on the argument, self-test capabilities are either turned on or off.

**Group**    Calibration and Diagnostic

**Syntax**    `DIAG:STATE {EXECute|ABORt}`

**Arguments**    `EXECute` starts diagnostics.  
`ABORt` stops diagnostics at the end of the current loop.

**Examples**    `DIAG:STATE EXECute` starts diagnostics.

## DISplay? (Query Only)

Returns the current display settings.

**Group**    Display

**Syntax**    `DISplay?`

## DISplay:CLOCK

Sets or returns whether the oscilloscope displays the date and time. The query form of this command returns an ON (1) or an OFF (0).

---

**NOTE.** For the MSO/DPO2000 series, the enumerations enable control of the date/time display. The query form will return 0 when the control is set to NONE or OFF and will return 1 when the control is set to DATETIME or ON.

---

**Group**    Display

**Syntax**    `DISplay:CLOCK {DATEOnly|TIMEOnly|DATETIME|NONE|ON|OFF|<NR1>}`  
`DISplay:CLOCK?`

**Related Commands**    [DATE](#), [TIME](#)

**Arguments**    DATEOnly enables the display of date.  
 TIMEOnly enables the display of time.  
 DATETIME or ON enables the display of both date and time.  
 NONE or OFF disables the display of date and time.  
 <NR1> = 0 disables the display of date and time; any other value enables the display of date and time.

**Examples**    DISPLAY:CLOCK ON enables display of date and time.  
 DISPLAY:CLOCK? might return :DISPLAY:CLOCK 1 indicating that the display of date and time is currently enabled.

## DISplay:DIGital:HEIght

Sets or returns the number of available digital waveform position slots.

---

**NOTE.** *If the number of live digital channels exceeds the number of slots, their height is reduced to a smaller size, allowing all digital channels to be displayed.*

---

**Group**    Display

**Syntax**    DISplay:DIGital:HEIght {SMAlL|MEDiUm|LARge}  
 DISplay:DIGital:HEIght?

**Arguments**    SMAlL sets the height to 40.  
 MEDiUm sets the height to 20.  
 LARge sets the height to 10.

## DISplay:FORMat

Sets or returns the display format.

**Group**    Display

**Syntax**    `DISplay:FORMat {YT|XY}`  
`DISplay:FORMat?`

**Arguments**    YT sets the display to a voltage versus time format and is the default mode.  
 XY argument displays one waveform against another. Selecting one source causes its corresponding source to be implicitly selected, producing a single trace from the two input waveforms.

**Examples**    `DISPLAY:FORMAT XY` sets the display format to XY.  
`DISPLAY:FORMAT?` might return `DISPLAY:FORMAT YT` indicating that the display format is YT.

## DISplay:GLITCh

Controls the display of the glitch capture background of the waveform. Set the intensity with [DISplay:INTENSITY:GLITCh](#).

**Group**    FilterVu

**Syntax**    `DISplay:GLITCh {ON|OFF|<NR1>}`  
`DISplay:GLITCh?`

**Related Commands**    [DISplay:INTENSITY:GLITCh](#)

**Arguments**    ON or `<NR1> ≠ 0` turns on the glitch capture waveform display.  
 OFF or `<NR1> = 0` turns off the glitch capture waveform display.

**Examples**    `DISPLAY:GLITCH?` might return `DISPLAY:GLITCH 1` indicating that there is glitch capture waveform display.  
`DISPLAY:GLITCH OFF` turns off the glitch capture waveform display.

## DISplay:GRATICule

Selects or queries the type of graticule the oscilloscope displays.

**Group**    Display

|                  |   |
|------------------|---|
| <b>Syntax</b>    | <code>DISPlay:GRaticule {CROSSHair FRame FULl GRId}</code><br><code>DISPlay:GRaticule?</code>   |
| <b>Arguments</b> | <code>CROSSHair</code> specifies a frame and cross hairs.<br><code>FRame</code> specifies a frame only.<br><code>FULl</code> specifies a frame, a grid and cross hairs.<br><code>GRId</code> specifies a frame and grid only.     |
| <b>Examples</b>  | <code>DISPLAY:GRATICULE FRame</code> sets the graticule type to display the frame only.<br><code>DISPLAY:GRATICULE?</code> might return <code>:DISPLAY:GRATICULE FULL</code> indicating that all graticule elements are selected. |

## DISPlay:INTENSITY? (Query Only)

Returns the display intensity settings.

|                 |   |
|-----------------|---|
| <b>Group</b>    | Display   |
| <b>Syntax</b>   | <code>DISPlay:INTENSITY?</code>   |
| <b>Examples</b> | <code>DISPLAY:INTENSITY?</code> might return <code>:DISPLAY:INTensity:WAVEFORM 35;GRATICULE 35;BACKLIGHT HIGH;GLITCH CAPTURE BACKGROUND INTENSITY 55</code> |

## DISPlay:INTENSITY:BACKLight

Sets and returns the waveform backlight intensity settings.

|                 |   |
|-----------------|---|
| <b>Group</b>    | Display   |
| <b>Syntax</b>   | <code>DISPlay:INTENSITY:BACKLight {LOW MEDIum HIGH}</code><br><code>DISPlay:INTENSITY:BACKLight?</code> |
| <b>Examples</b> | <code>DISPLAY:INTENSITY:BACKLIGHT?</code> might return <code>:DISPLAY:INTensity:BACKLIGHT HIGH</code>   |

## DISplay:INTENSITY:GLITCh

Sets the intensity of the glitch capture background of the waveform display. The intensity can be set from 5 to 100% in increments of 5%. The intensity of the glitch capture background can only be adjusted when the glitch capture background is displayed and FilterVu filtering is in use. Otherwise, the intensity is fixed.

Use [DISplay:GLITCh](#) to turn the glitch capture background ON and OFF.

Use [FILTERVu:FREQuency](#) to enable filtering. Filtering is enabled if the frequency chosen is different than the oscilloscope's full bandwidth.

**Group** FilterVu

**Syntax** DISplay:INTENSITY:GLITCh {<NR1>}

### Related Commands

**Arguments** <NR1> is the glitch intensity and ranges from 5 to 100 percent.

**Examples** DISPLAY:INTENSITY:GLITCH ? might return DISPLAY:INTENSITY:GLITCH 5 indicating that the glitch intensity is 5%.

## DISplay:INTENSITY:GRAticule

Sets and returns the display graticule intensity settings.

**Group** Display

**Syntax** DISplay:INTENSITY:GRAticule <NR1>  
DISplay:INTENSITY:GRAticule?

**Arguments** <NR1> is the graticule intensity and ranges from 0 to 100 percent.

**Examples** DISPLAY:INTENSITY:GRATICULE? might return  
DISPLAY:INTENSITY:GRATICULE 30

## DISplay:INTENSITY:WAVEform

Sets and returns the display waveform intensity settings.

|                  |   |
|------------------|---|
| <b>Group</b>     | Display   |
| <b>Syntax</b>    | DISPlay:INTENSITY:WAVEform <NR1><br>DISPlay:INTENSITY:WAVEform?   |
| <b>Arguments</b> | <NR1> is the waveform intensity and ranges from 1 to 100 percent.   |
| <b>Examples</b>  | DISPLAY:INTENSITY:WAVEFORM? might return<br>DISPLAY:INTENSITY:WAVEFORM 60<br><br>as the intensity of the waveforms. |

## DISPlay:PERsistence

Sets or returns the display persistence. This affects the display only.

---

**NOTE.** *When Persistence is set to Infinite, it does not mean that the brightness of any pixel should never decrease. The brightness of a pixel is proportionally dependent on the ratio between its intensity (which does NOT decrease at Infinite Persistence) and the maximum value of intensity of any pixel on the screen. If a particular pixel get hit less often than others, its brightness will decrease over time. It will become less bright relative to the pixels that get hit often.*

---

|                  |  |
|------------------|--|
| <b>Group</b>     | Display  |
| <b>Syntax</b>    | DISPlay:PERsistence {<NR3> CLEAR AUTO MINIMUM INFINITE}<br>DISPlay:PERsistence?  |
| <b>Arguments</b> | <NR3> specifies the time of the persistence.<br><br>CLEAR resets the persist time count down and clears the display of acquired points.<br><br>INFINITE displays waveform points until a control change resets the acquisition system. When persistence is set to infinite, it does not mean that the brightness of any pixel should never decrease. The brightness of a pixel is proportionally dependent on the ratio between its intensity, which does NOT decrease at infinite persistence, and the maximum value of intensity of any pixel on the screen. Thus, if a particular pixel gets hit less often than others, its brightness will decrease over time. It will become less bright relative to the pixels that get hit often.<br><br>AUTO specifies that the oscilloscope automatically determines the best waveform persistence based on the value of waveform intensity (DISPLAY:INTENSITY:WAVEFORM) |

MINIMUM specifies that the waveform persistence is set to the minimum value of 0.0E0.

**Examples** DISPLAY:PERSISTENCE 3 specifies that the waveform points are displayed fading for 3 seconds before they completely disappear.

## D<x>

Sets or returns parameters for digital channel <x>, where x is the channel number.

**Group** MSO

**Syntax** D<x>

## D<x>:LABel

Sets or returns the waveform label for digital channel <x>, where x is the channel number.

**Group** MSO

**Syntax** D<x>:LABel <Qstring>  
D<x>:LABel?

**Arguments** <Qstring> is an alpha-numeric string of text, enclosed in quotes, that contains the text label information for the digital channel <x> waveform. The text string is limited to 30 characters.

**Examples** D4:LABEL? might return :D4:LABEL "Clk wfm 4".

## D<x>:POStion

Sets or returns the vertical position for digital channel <x>, where x is the channel number.

---

**NOTE.** Unlike analog channels, which use the center baseline as the zero point, digital channels use the bottom logic level as the zero point. Because of the slot model used to describe digital channel position, the oscilloscope rounds off divisions to the nearest slot.

---



|                  |   |
|------------------|---|
| <b>Group</b>     | MSO   |
| <b>Syntax</b>    | D<x>:POSition <NR3><br>D<x>:POSition?   |
| <b>Arguments</b> | <NR3> is the position of the digital channel in slot units. Use the oscilloscope front-panel controls to place the channel; then query the channel to obtain an exact value for the position. |

## D<x>:THREshold

Sets or returns the logical threshold for digital channel <x>, where x is the channel number.

|                  |  |
|------------------|--|
| <b>Group</b>     | MSO  |
| <b>Syntax</b>    | D<x>:THREshold {ECL TTL <NR3>}<br>D<x>:THREshold?  |
| <b>Arguments</b> | ECL sets the digital threshold for channel <x> to a preset ECL high level of -1.3V.<br>TTL sets the digital threshold for channel <x> to a preset TTL high level of 1.4V.<br><NR3> specifies the digital threshold for channel <x> in volts. |
| <b>Examples</b>  | D5:THRESHOLD ECL sets the threshold for D5 digital channel to ECL levels.<br>D14:threshold? might return :D14:THRESHOLD 1.2500 indicating that the threshold level for the D14 digital channel is 1.25 volts.                                |

## \*ESE

Sets and queries the bits in the Event Status Enable Register (ESER). The ESER prevents events from being reported to the Status Byte Register (STB). For a detailed discussion on how to use registers, see Registers.

|               |                     |
|---------------|---------------------|
| <b>Group</b>  | Status and Error    |
| <b>Syntax</b> | *ESE <NR1><br>*ESE? |

**Related Commands**    [\\*CLS](#), [DESE](#), [\\*ESR?](#), [EVENT?](#), [EVMsg?](#), [\\*SRE](#), [\\*STB?](#)

**Arguments**    <NR1> specifies the binary bits of the ESER according to this value, which ranges from 0 through 255.

The power-on default for the ESER is 0 if \*PSC is 1. If \*PSC is 0, the ESER maintains the previous power cycle value through the current power cycle.

---

**NOTE.** *Setting the DESER and the ESER to the same values allows only those codes to be entered into the Event Queue and summarized on the ESB bit (bit 5) of the Status Byte Register. Use the DESE command to set the DESER.*

---

**Examples**    \*ESE 209 sets the ESER to binary 11010001, which enables the PON, URQ, EXE, and OPC bits.

\*ESE? might return 186, showing that the ESER contains the binary value 10111010.

## \*ESR? (Query Only)

Returns the contents of the Standard Event Status Register (SESR). \*ESR? also clears the SESR (reading the SESR clears it). For a detailed discussion on how to use registers, see Registers.

**Group**    Status and Error

**Syntax**    \*ESR?

**Related Commands**    [ALLEv?](#), [\\*CLS](#), [DESE](#), [\\*ESE](#), [EVENT?](#), [EVMsg?](#), [\\*SRE](#), [\\*STB?](#)

**Examples**    \*ESR? might return 213, showing that the SESR contains the binary value 11010101.

## ETHERnet:DHCPbootp

Sets or returns the network initialization search for a DHCP/BOOTP server.

**Group**    Ethernet

|                  |  |
|------------------|--|
| <b>Syntax</b>    | ETHERnet:DHCPbootp {ON OFF}<br>ETHERnet:DHCPbootp?   |
| <b>Arguments</b> | ON enables the oscilloscope to search the network for a DHCP or BOOTP server in order to automatically assign a dynamic IP address to the oscilloscope.  |
|                  | <hr/> <b>NOTE.</b> <i>Do not use DHCP/BOOTP searching if your oscilloscope has been assigned a static address on a network. If you set this command to ON, the DHCP/BOOTP search will delete or change your static IP address information.</i> <hr/> |
|                  | OFF disables the oscilloscope to search the network for a DHCP or BOOTP server.  |
| <b>Examples</b>  | ETHERNET:DHCPBOOTP ON sets the oscilloscope to search for a DHCP or BOOTP server and assign a dynamic IP address to the oscilloscope.  |

## ETHERnet:DNS:IPADDRESS

Sets or returns the network Domain Name Server (Dns) IP address.

|                  |   |
|------------------|---|
| <b>Group</b>     | Ethernet  |
| <b>Syntax</b>    | ETHERnet:DNS:IPADDRESS <QString><br>ETHERnet:DNS:IPADDRESS?   |
| <b>Arguments</b> | <QString> is a standard IP address value, enclosed in quotes.   |
| <b>Examples</b>  | ETHERNET:DNS:IPADDRESS "128.196.13.352" sets the Dns IP address that the oscilloscope uses to communicate with the network. |

## ETHERnet:DOMAINname

Sets or returns the network domain name.

|               |   |
|---------------|---|
| <b>Group</b>  | Ethernet  |
| <b>Syntax</b> | ETHERnet:DOMAINname <Qstring><br>ETHERnet:DOMAINname? |

**Arguments** <QString> is the network domain name, enclosed in quotes.

**Examples** ETHERNET:DOMAINNAME "Alpha1.Mycorp.com" sets the domain name that the oscilloscope uses to communicate with the network.

## ETHERnet:ENET:ADDRESS? (Query Only)

Returns the Ethernet address value assigned to the oscilloscope. This is assigned at the factory and can not be changed.

**Group** Ethernet

**Syntax** ETHERnet:ENET:ADDRESS?

**Examples** ETHERNET:ENET:ADDRESS? returns an Ethernet address such as 08:00:11:01:02:03

## ETHERnet:GATEWay:IPADDRESS

Sets or returns the remote interface gateway IP address.

**Group** Ethernet

**Syntax** ETHERnet:GATEWay:IPADDRESS <QString>  
ETHERnet:GATEWay:IPADDRESS?

**Arguments** <QString> is a standard IP address value, enclosed in quotes.

**Examples** ETHERNET:GATEWAY:IPADDRESS "128.143.16.1" sets the gateway IP address.

## ETHERnet:HTTPPort

Sets or returns the remote interface HTTP port value.

**Group** Ethernet

**Syntax**   ETHERnet:HTTPPort <QString>  
ETHERnet:HTTPPort?

**Arguments**   <QString> is an integer port number, enclosed in quotes.

---

**NOTE.** Consider the following if you are using the e\*Scope™ control software. If you don't enter a port address in the URL, then the ETHERnet:HTTPPort value must be set to "80", which is the default port for HTTP protocol. If you use a URL with a port address (for example: http://DPO2004-04WKL4:1234), the port number is specified by the number after the colon. Set the ETHERnet:HTTPPort value to this same number.

---

**Examples**   ETHERNET:HTTPPORT "80" sets the HTTP port value to 80.

## ETHERnet:IPADDRESS

Sets or returns the IP address assigned to the oscilloscope.

**Group**   Ethernet

**Syntax**   ETHERnet:IPADDRESS <QString>  
ETHERnet:IPADDRESS?

**Arguments**   <QString> is a standard IP address value, enclosed in quotes.

**Examples**   ETHERNET:IPADDRESS "123.121.13.214" sets the oscilloscope's IP address.

## ETHERnet:NAME

Sets or returns the network name assigned to the oscilloscope.

**Group**   Ethernet

**Syntax**   ETHERnet:NAME <QString>  
ETHERnet:NAME?

**Arguments**   <QString> is the network name assigned to the oscilloscope, enclosed in quotes.

**Examples**     ETHERNET:NAME "labcscope1" sets the oscilloscope's network name.

## ETHERnet:PASSWord

Sets or returns the HTTP Ethernet access password. If a password is set, the user must enter the password before the user's Web browser can access the oscilloscope.

**Group**     Ethernet

**Syntax**     ETHERnet:PASSWord <new>  
ETHERnet:PASSWord?

**Arguments**     <new> is a new password, enclosed in quotes.

**Examples**     ETHERNET:PASSWORD "ZEN53" replaces the current Ethernet password with the new password ZEN53.

ETHERNET:PASSWORD? might return :ETHERNET:PASSWORD "ZEN53".

## ETHERnet:PING (No Query Form)

Causes the oscilloscope to ping the gateway IP address.

**Group**     Ethernet

**Syntax**     ETHERnet:PING EXECute

**Examples**     ETHERNET:PING EXECUTE causes the oscilloscope to ping the gateway IP address.

## ETHERnet:PING:STATUS? (Query Only)

Returns the results from sending the [ETHERnet:PING](#) command to ping the gateway IP address.

**Group**     Ethernet

**Syntax**    ETHERnet:PING:STATUS?

**Returns**    OK is returned if the computer at the gateway IP address answers.  
 NORESPONSE is returned if the computer at the gateway IP address does not answer.  
 INPROGRESS is returned if the ping operation is still executing.

## ETHERnet:SUBNETMask

Sets or returns the remote interface subnet mask value.

**Group**    Ethernet

**Syntax**    ETHERnet:SUBNETMask <QString>  
 ETHERnet:SUBNETMask?

**Arguments**    <QString> is the subnet mask value, enclosed in quotes.

**Examples**    ETHERNET:SUBNETMASK "255.255.255.0" sets the subnet mask value using standard IP address notation format.

## EVENT? (Query Only)

Returns an event code from the Event Queue that provides information about the results of the last \*ESR? read. EVENT? also removes the returned value from the Event Queue.

**Group**    Status and Error

**Syntax**    EVENT?

**Related Commands**    ALLEv?, \*CLS, DESE, \*ESE, \*ESR?, EVMsg?, \*SRE, \*STB?

**Examples**    EVENT? might return :EVENT 110, showing that there was an error in a command header. (See page 3-12, *Messages*.)

## EVMsg? (Query Only)

Removes a single event code from the Event Queue that is associated with the results of the last [\\*ESR?](#) read and returns the event code along with an explanatory message. For information, see Event Handling.

**Group** Status and Error

**Syntax** EVMsg?

**Related Commands** [ALLEv?](#)  
[\\*CLS](#), [DESE](#), [\\*ESE](#), [\\*ESR?](#), [EVENT?](#), [\\*SRE](#), [\\*STB?](#)

**Returns** The event code and message in the following format:  
 <Event Code><Comma><QString>[<Event Code><Comma><QString>...]<QString>::= <Message>;[<Command>] where <Command> is the command that caused the error and may be returned when a command error is detected by the oscilloscope. As much of the command will be returned as possible without exceeding the 60 character limit of the <Message> and <Command> string combined. The command string is right-justified.

**Examples** EVMSG? might return :EVMSG 110,"Command header error".

## EVQty? (Query Only)

Returns the number of event codes in the Event Queue. This is useful when using the [ALLEv?](#) query, which returns the exact number of events.

**Group** Status and Error

**Syntax** EVQty?

**Related Commands** [ALLEv?](#), [EVENT?](#), [EVMsg?](#)

**Examples** EVQTY? might return :EVQTY 3, indicating that the number of event codes in the Event Queue.



## FACTory (No Query Form)

Resets the oscilloscope to its factory default settings.

This command does the following:

- Clears the Event Status Enable Register
- Clears the Service Request Enable Register
- Sets the Device Event Status Enable Register to 255
- Purges all defined aliases
- Enables all Command Headers
- Sets the macro defined by \*DDT to a "zero-length field"
- Clears the pending operation flag and associated operations

This command does not reset the following:

- Communication settings
- State of the VXI-11 (Ethernet IEEE Std 488.2) interface.
- Calibration data that affects device specifications
- Protected user data
- Stored settings
- Power On Status Clear Flag
- Oscilloscope password

**Group** Save and Recall

**Syntax** FACTory

**Related Commands** [\\*PSC](#), [\\*RCL](#), [RECALL:SETUp](#), [\\*RST](#), [\\*SAV](#), [SAVe:SETUp](#)

**Arguments** None

**Examples** FACTORY resets the oscilloscope to its factory default settings.

## FILESystem

Returns the directory listing of the current working directory and the number of bytes of free space available. This query is the same as the [FILESystem:DIR?](#) query and the [FILESystem:FREEspace?](#) query.

**Group** File System

**Syntax** FILESystem  
FILESystem?

**Related Commands** [FILESystem:CWD](#), [FILESystem:DELEte](#), [FILESystem:DIR?](#), [FILESystem:REName](#)

**Arguments** None

**Examples** FILESYSTEM? might return  
  
: FILESYSTEM:DIR "TEK00000.BMP", "GLITCH1.PNG", "TEMP.TMP",  
"FILE1.WFM", "FILE2.WFM", "MATH1.WFM", " REF1.WFM", "REF2.WFM"

## FILESystem:CWD

Sets or returns the current working directory (CWD) for FILESystem commands. The default working directory is "E:/". Anytime you use this command to change the directory, the directory that you specify is retained as the current working directory until you either change the directory or you delete the directory. If you delete the current working directory, the oscilloscope resets current working directory to the default directory (E:) the next time the oscilloscope is powered on or the next time you execute a file system command.

This command supports the permutations of file and directory names supported by Microsoft Windows:

- Relative path names; for example, "./temp"
- Absolute path names; for example, "E:/Wfms"
- Implied relative path names; for example "NEWFILE.TXT" becomes "E:/TEKSCOPE/NEWFILE.TXT" if the current working directory is "E:/TEKSCOPE"

**Group** File System

|                  |   |
|------------------|---|
| <b>Syntax</b>    | <code>FILESystem:CWD {&lt;new working directory path&gt;}</code>  |
| <b>Arguments</b> | <new working directory path> is a quoted string that defines the current working; a directory name can have up to 8 characters with an extension of up to 3 characters.   |
| <b>Examples</b>  | <p><code>FILESYSTEM:CWD "E:/TEKSCOPE/IMAGES"</code> sets the current working directory to images.</p> <p><code>FILESYSTEM:CWD?</code> might return</p> <p><code>:FILESYSTEM:CWD "E:/TEKSCOPE/WAVEFORMS"</code> indicating that the current working directory is set to waveforms.</p> |

## FILESystem:DELEte (No Query Form)

This command deletes a named file. If you specify a directory name, it will delete the directory and all of its contents, the same as the RMDir command. You can also specify the filename as \*.\* to delete all of the files in the current or specified directory.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | File System  |
| <b>Syntax</b>           | <code>FILESystem:DELEte &lt;file path&gt;</code>   |
| <b>Related Commands</b> | <p><a href="#">FILESystem:CWD</a></p> <p><a href="#">FILESystem:RMDir</a></p>  |
| <b>Arguments</b>        | <p>&lt;file path&gt; is a quoted string that defines the file name and path. If the file path is within the current working directory, you need only specify the file name.</p> <p>The argument *.* will delete all files and subdirectories within the current working directory.</p> |
| <b>Examples</b>         | <code>FILESYSTEM:DELETE "NOT_MINE.SET"</code> deletes the file named NOT_MINE.SET from the current working directory.  |

## FILESystem:DIR? (Query Only)

Returns a list of quoted strings. Each string contains the name of a file or directory in the current working directory.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | File System   |
| <b>Syntax</b>           | FILESystem:DIR?   |
| <b>Related Commands</b> | <a href="#">FILESystem:CWD</a> , <a href="#">FILESystem:MKDir</a>   |
| <b>Arguments</b>        | None  |
| <b>Returns</b>          | FILESystem:DIR? returns a list of files and directories in the current working directory.   |
| <b>Examples</b>         | <pre> FILESYSTEM:DIR? might return : FILESYSTEM:DIR "TEK00000.PNG", "CANSETUP.SET", "WFM1.ISF", "MYIMAGES"                     </pre> |

## FILESystem:FORMat (No Query Form)

Formats a mass storage device. This command should be used with extreme caution as it causes all data on the specified mass storage device to be lost. Drive letters (e.g., E:) are case sensitive and must be upper case. For all other FILESYSTEM commands, drives letters are not case sensitive. Example: FILE:FORMAT "E:/" Formats the USB flash drive installed in the oscilloscope's front panel USB port.

|                  |   |
|------------------|---|
| <b>Group</b>     | File System   |
| <b>Syntax</b>    | FILESystem:FORMat   |
| <b>Arguments</b> | <drive name> is a quoted string that defines the drive to format.   |
| <b>Examples</b>  | <pre> FILESYSTEM:FORMAT "E:/"                     </pre> <p>Formats the USB flash drive installed in the oscilloscope's front panel USB port.</p> |

## FILESystem:FREEspace? (Query Only)

Returns the number of bytes of free space on the current drive.

**Group** File System

**Syntax** FILESystem:FREESpace?

**Related Commands** [FILESystem:FREESpace?](#), [FILESystem:CWD](#)

## FILESystem:MKDir (No Query Form)

Creates a new folder.

**Group** File System

**Syntax** FILESystem:MKDir <directory path>

**Related Commands** [FILESystem:CWD](#), [FILESystem:DIR?](#)

**Arguments** <directory path> is a quoted string that specifies the directory to create

**Examples** FILESYSTEM:MKDIR "E:/NewDirectory" creates the directory named *NEWDIRECTORY* at the root of the E drive.

These two commands create the directory MYNEWSUBDIRECTORY within the existing directory mydirectory at the root of the E drive:

```
FILESYSTEM:CWD "E:/MyDirectory";:FILESYSTEM:MKDIR
"myNewSubDirectory"
```

This, of course, assumes that *E:/MYDIRECTORY* already existed and was not a read-only directory.

## FILESystem:READFile (No Query Form)

Writes the contents of the specified file to the specified interface. If the file does not exist or is not readable, an appropriate error event is posted.

**Group** File System

**Syntax** FILESystem:READFile <QString>

**Related Commands**    [FILESystem:CWD](#)

**Arguments**    <Qstring> is a quoted string that defines the file name and path. If the file path is within the current working directory, specify only the file name.

**Examples**    FILESYSTEM:READFILE "E:/TEST\_DATA/TEK00016CH1.CSV" reads the content of the specified file, if the file exists and is readable, and sends the content of the file to the current interface.

## FILESystem:REName (No Query Form)

Assigns a new name to an existing file.

**Group**    File System

**Syntax**    FILESystem:REName <old file path>,<new file path>

**Related Commands**    [FILESystem:CWD](#)

**Arguments**    <old file path> is a quoted string that defines the file name and path. If the file path is within the current working directory, you need only specify the file name.  
  
                  <new file path> is a quoted string that defines the file name and path. If the file path is within the current working directory, you need only specify the file name.

**Examples**    FILESYSTEM:RENAME "E:/TEK00000.SET", "D:/MYSETTING.SET" gives the file named TEK00000.SET the new name of MYSETTING.SET. The file remains in the root directory on the D drive.

## FILESystem:RMDir (No Query Form)

Deletes a named directory. This command deletes the specified directory and all of its contents. The directory must not be a read-only directory.

**Group**    File System

**Syntax**    FILESystem:RMDir <directory path>

**Related Commands** [FILESystem:CWD](#)

**Arguments** `<directory path>` is a quoted string that defines the directory name and path. If the file path is within the current working directory, you need only specify the file name.

**Examples** `FILESYSTEM:RMDIR "E:/OldDirectory"` removes the directory named `olddirectory` from the root of the E drive.

## FILESystem:WRITEFile (No Query Form)

Writes the specified block data to a file in the oscilloscope current working directory. If the specified file does not exist or is not readable, an appropriate error event is posted. The maximum length of the block data is 262144 bytes.

**Group** File System

**Syntax** `FILESystem:WRITEFile <file path>, <data>`

**Related Commands** [FILESystem:CWD](#)

**Arguments** `<file path>` is the quoted string that defines the file name and path. If the path is within the current working directory, specify the file name only.

`<data>` can be either DEFINITE LENGTH encoding or INDEFINITE LENGTH ARBITRARY BLOCK PROGRAM DATA encoding as described in IEEE488.2.

## FILTERVu:FREQuency

Sets or queries the FilterVu frequency to the closest value supported with the current acquisition settings.

**Group** FilterVu

**Syntax** `FILTERVu:FREQuency?`  
`FILTERVu:FREQuency <NR3>`

**Related Commands** [FILTERVu:FREQuency:AVAILable?](#)

**Arguments** <NR3> is the integer format for the current FilterVu frequency setting.

**Examples** FILTERVU:FREQUENCY? might return FILTERVU:FREQUENCY 100000000 indicating that the FilterVu frequency is 100 MHz.

## FILTERVu:FREQuency:AVAILable? (Query Only)

Returns a comma separated list of filter frequencies available based on the current instrument settings. The source waveform (as specified by the [DATA:SOURce](#)) must be turned on for this query to generate a response.

**Group** FilterVu

**Syntax** FILTERVu:FREQuency:AVAILable?

**Related Commands** [FILTERVu:FREQuency](#)

**Examples** FILTERVU:FREQUENCY:AVAILABLE? might return FILTERVU:FREQUENCY:AVAILABLE 200000000,150000000,75000000,42000000,21000000 indicating that the available FilterVu frequencies are 200 MHz, 150 MHz, 75 MHz, 42 MHz and 21 MHz.

## FPAnel:PRESS (No Query Form)

Simulates the action of pressing a specified front-panel button.

When the front panel is locked, the front-panel button and multipurpose knob operations are suspended. The `FPAnel:PRESS` and the [FPAnel:TURN](#) commands will also not work. You can work around this by using the appropriate programmatic interface commands, instead of the front-panel commands.

**Group** Miscellaneous

**Syntax** FPAnel:PRESS <button>

**Arguments** <button> is the name of a front-panel button. Most of the argument names associate directly with their front panel buttons. For example, `AUTOSet` is for the Autoset button. The <button> enumeration arguments and their associations with the front panel buttons are listed below.



**Table 2-37: FPanel:PRESS arguments**

| <b>Argument</b> | <b>Description</b>                  |
|-----------------|-------------------------------------|
| ACQuire         | <b>Acquire</b> button               |
| SAVERecall      | Save/Recall <b>Menu</b> button      |
| MEASurement     | <b>Measure</b> button               |
| UTILity         | <b>Utility</b> button               |
| SEArch          | <b>Search</b> button                |
| TESt            | <b>Test</b> button                  |
| MATh            | <b>M</b> button                     |
| REF             | <b>R</b> button                     |
| TRIGger         | Trigger <b>Menu</b> button          |
| CH1             | Channel1 select button              |
| CH2             | Channel2 select button              |
| CH3             | Channel2 select button              |
| CH4             | Channel4 select button              |
| B1              | Bus1 select button                  |
| B2              | Bus2 select button                  |
| D15D0           | Digital channels select buttons     |
| DEFaultsetup    | <b>Default Setup</b> button         |
| MENUOff         | <b>Menu Off</b> button              |
| SElect          | <b>Select</b> button                |
| FINE            | <b>Fine</b> button                  |
| CURsor          | <b>Cursors</b> button               |
| INTensity       | <b>Intensity</b> button             |
| MARK            | Mark <b>Set/Clear</b> button        |
| PREv            | Previous arrow button               |
| PAUse           | Play/pause button                   |
| NEXt            | Next arrow button                   |
| RUnstop         | <b>Run/Stop</b> button              |
| SINGleseq       | <b>Single</b> button                |
| AUTOset         | <b>Autoset</b> button               |
| SETTO50         | Set Trigger to 50%                  |
| FORCetrig       | <b>Force Trig</b> button            |
| HARDcopy        | Hardcopy button                     |
| SAVEBUtton      | <b>Save</b> button                  |
| BMENU1          | Screen left-most bottom menu button |
| BMENU2          | Screen bottom menu button           |
| BMENU3          | Screen bottom menu button           |
| BMENU4          | Screen bottom menu button           |

**Table 2-37: FPanel:PRESS arguments (cont.)**

| <b>Argument</b>             | <b>Description</b>                            |
|-----------------------------|---|
| BMENU5                      | Screen bottom menu button                     |
| BMENU6                      | Screen bottom menu button                     |
| BMENU7                      | Screen right-most bottom menu button          |
| RMENU1                      | Screen top-most side menu button              |
| RMENU2                      | Screen side menu button                       |
| RMENU3                      | Screen side menu button                       |
| RMENU4                      | Screen side menu button                       |
| RMENU5                      | Screen bottom-most side menu button           |
| MAGnify                     | Magnify (zoom) button (not the zoom/pan knob) |
| FITERVu (2000 series only)  | FilterVu frequency button                     |
| WFMOOnly (2000 series only) | Waveform data button                          |

**Examples**    `FPanel:PRESS AUTOSET` executes the oscilloscope Autoset function.

## FPanel:TURN (No Query Form)

Simulates the action of turning a specified front-panel control knob.

When the front panel is locked, the front-panel button and multipurpose knob operations are suspended. The [FPanel:PRESS](#) and `FPanel:TURN` commands will also not work, and, they will not generate an error. You can work around this by using the appropriate programmatic interface commands, instead of the front-panel commands. For example, to set the trigger level to 50%, you could use `TRIGGER:A SETLevel`. To force a trigger, you could use `TRIGGER FORCe`.

**Group**    Miscellaneous

**Syntax**    `FPanel:TURN <knob>, [<n>]`

**Arguments**    `<knob>` is the name of a rotating control.

A comma (,) separates the control knob argument from the numeric optional rotation value argument. In the absence of the numeric rotation value argument, the default is 1 (clockwise). You do not need a white space between the arguments and the comma.

`<n>` represents the rotation direction and magnitude of rotation. Negative values represent a counterclockwise knob rotation, and positive values represent a

clockwise rotation. The magnitude of <n> specifies the amount of the turn, where <n> = 1 represents turning the knob one unit, <n> = 2 represents turning the knob two units, <n> = 4 represents turning the knob four units, and so on. The range of units depends on which front panel knob is specified.

**Table 2-38: FPanel:TURN arguments**

| Argument     | Description                     |
|--------------|---------------------------------|
| GPKNOB1      | <b>Multipurpose a</b> knob      |
| GPKNOB2      | <b>Multipurpose b</b> knob      |
| HORZPos      | Horizontal <b>Position</b> knob |
| HORZScale    | Horizontal <b>Scale</b> knob    |
| PANKNOB1     | Outer pan knob                  |
| TRIGLevel    | Trigger <b>Level</b> knob       |
| VERTPOS<n>   | Vertical <b>Position</b> knob   |
| VERTSCALE<n> | Vertical <b>Scale</b> knob      |
| ZOOM         | Inner zoom knob                 |

**Examples**    FPANEL :TURN TRIGLEVEL ,10 duplicates turning the front-panel Trigger Level knob clockwise by 10 units.

## GPIBUsb:ID? (Query Only)

Returns the identification string of the connected TEK-USB-488 adaptor module and firmware version. If a TEK-USB-488.2 module is not connected, the system returns “Not detected”.

**Group**    Miscellaneous

**Syntax**    GPIBUsb:ID?

## HARDCopy (No Query Form)

Sends a hard copy of the screen display to the currently active printer using the current palette and layout settings.

**Group**    Hard Copy

**Syntax**    HARDCopy {START}  
HARDCopy?

**Related Commands**    \*WAI, \*CLS

**Arguments**    START sends a block of data representing the current screen image to the requested port. The data sent is in the image format specified by [SAVE:IMAGe:FILEFormat](#), and the compression level is controlled by whatever format has been selected (BMP and TIFF are uncompressed, while PNG is compressed).

**Examples**    HARDCOPY initiates a screen copy to the active printer.

## HARDCopy:INKSaver

Changes hard copy output to print traces and graticule on a white background while retaining waveform color information (except for channel 1, which prints as dark blue because yellow does not show up well and is difficult to see on a white background). This option can significantly reduce print time and quantities of ink required compared with WYSIWYG dark background images.

**Group**    Hard Copy

**Syntax**    HARDCopy:INKSaver?

**Arguments**    ON or <NR1> ≠ 0 sets the ink saver mode on.  
OFF or <NR1> = 0 sets the ink saver mode off.

**Examples**    HARDCOPY:INKSAVER ON will cause subsequent hard copy output to display the screen on a white background.

## HARDCopy:PREVIEW (No Query Form)

Displays a preview of the current screen contents with the InkSaver palette applied.

**Group**    Hard Copy

**Syntax**    HARDCopy:PREVIEW {ON|OFF|<NR1>}

**Arguments** ON or <NR1> ≠ 0 turns preview mode on.  
 OFF or <NR1> = 0 turns preview mode off.

## HEADer

Sets or returns the Response Header Enable State that causes the oscilloscope to either include or omit headers on query responses.

---

**NOTE.** *This command does not affect IEEE Std 488.2-1987 Common Commands (those starting with an asterisk); these commands never return headers. This command does affect the Response Header Enable State of both the USBTMC and VXI-11 interfaces. Refer to the Introduction for additional information.*

---

**Group** Miscellaneous

**Syntax** HEADer {OFF|ON|<NR1>}  
 HEADer?

**Related Commands** [VERBose](#)

**Arguments** OFF sets the Response Header Enable State to false. This causes the oscilloscope to omit headers on query responses, so that only the argument is returned.

ON sets the Response Header Enable State to true. This causes the oscilloscope to include headers on applicable query responses. You can then use the query response as a command.

<NR1> = 0 sets the Response Header Enable State to false; any other value sets this state to true.

**Examples** HEADer OFF specifies that the oscilloscope omits headers on query responses, so that only the argument is returned.

HEADer? might return :HEADer 1 indicating that the oscilloscope is including headers on applicable query responses.

## HORizontal? (Query Only)

Returns all settings for the horizontal commands.

**Group** Horizontal

**Syntax**    `HORizontal?`

**Examples**    `HORIZONTAL?` might return the following horizontal settings  
`:HORIZONTAL:DIGITAL:RECORDLENGTH:MAIN 100.0000E+3;`  
`:HORIZONTAL:DIGITAL:SAMPLERATE:MAIN 1.0000E+9;`  
`:HORIZONTAL:MAIN:UNITS "s";`  
`:HORIZONTAL:POSITION 50.0000;`  
`SAMPLERATE 1.0000E+9;`  
`SCALE 4.0000E-6;`  
`RECORDLENGTH 100000;`  
`DELAY:MODE 1;`  
`TIME 0.0E+0`

## **HORizontal:ACQLENGTH? (Query Only)**

Returns the record length.

**Group**    Horizontal

**Syntax**    `HORizontal:ACQLENGTH?`

**Related Commands**    [HORizontal:RECOrdlength](#)

**Examples**    `HORIZONTAL:ACQLENGTH?` might return `HORizontal:ACQLENGTH? :HORIZONTAL:ACQLENGTH 1.0000E+6` indicating that the record length is 1 million points.

## **HORizontal:DELAy:MODE**

Sets or returns the horizontal delay mode.

**Group**    Horizontal

**Syntax**    `HORizontal:DELAy:MODE {OFF|ON|<NR1>}`  
`HORizontal:DELAy:MODE?`

**Related Commands**    [HORizontal:POSition](#)

**Arguments**    `OFF` sets the Horizontal Delay Mode to off. This causes the [HORizontal:POSition](#) command to horizontally position the waveform.

`ON` sets the Horizontal Delay Mode to on. This causes the [HORizontal:DELAy:TIME](#) command to horizontally position the waveform.

<NR1> = 0 sets the Horizontal Delay Mode to off; any other value sets this mode to on.

**Examples** HORIZONTAL:DELAY:MODE OFF sets the Horizontal Delay Mode to off, allowing the [HORizontal:POSition](#) command to horizontally position the waveform.

HORIZONTAL:DELAY:MODE? might return HORIZONTAL:DELAY:MODE OFF indicating that the [HORizontal:POSition](#) command horizontally positions the waveform.

## HORizontal:DELay:TIME

Sets or returns the horizontal delay time. The amount of time the acquisition is delayed depends on sample rate and record length.

**Group** Horizontal

**Syntax** HORIZONTAL:DELAY:TIME <NR3>  
HORIZONTAL:DELAY:TIME?

**Arguments** NR3 is the delay in seconds.

**Examples** HORIZONTAL:DELAY:TIME 0.3 sets the delay of acquisition data so that the resulting waveform is centered 300 ms after the trigger occurs.

## HORizontal:DIGital:RECOrdlength:MAIN? (Query Only)

Returns the record length of the main digital acquisition.

**Group** Horizontal

**Syntax** HORIZONTAL:DIGITAL:RECORDLENGTH:MAIN?

## HORizontal:DIGital:SAMPLERate:MAIN? (Query Only)

Returns the sample rate of the main digital acquisition.

**Group** Horizontal

**Syntax**    `HORizontal:DIGital:SAMPLERate:MAIN?`

## HORizontal:POSition

Sets or returns the horizontal position. If Horizontal Delay Mode is turned off, this command is equivalent to adjusting the HORIZONTAL POSITION knob on the front panel. When Horizontal Delay Mode is on, this command stores a new horizontal position that is used when Horizontal Delay Mode is turned off.

**Group**    Horizontal

**Syntax**    `HORizontal:POSition <NR3>`  
`HORizontal:POSition?`

**Arguments**    `<NR3>` is the horizontal position expressed as the percentage of the waveform displayed left of the center of the graticule.

**Examples**    `HORIZONTAL:POSITION 50` sets the horizontal position to 50%.  
`HORIZONTAL:POSITION?` might return `:HORIZONTAL:POSITION 100` indicating that the horizontal position is set to 100%.

## HORizontal:PREViewstate? (Query Only)

Returns a boolean value to indicate whether the acquisition system is in the preview state.

**Group**    Horizontal

**Syntax**    `HORizontal:PREViewstate?`

**Returns**    `<NR1> = 1` if the acquisition system is in the preview state.  
`<NR1> = 0` if the acquisition system is not in the preview state.

## HORizontal:RECOrdlength

Sets the horizontal record length of acquired waveforms. The query form of this command returns the current horizontal record length.



|                  |   |
|------------------|---|
| <b>Group</b>     | Horizontal  |
| <b>Syntax</b>    | <code>HORIZONTAL:RECORDlength &lt;NR1&gt;</code><br><code>HORIZONTAL:RECORDlength?</code>   |
| <b>Arguments</b> | <NR1> represents the supported values for horizontal record lengths, which are: 100000 or 1000000.  |
| <b>Examples</b>  | <code>HORIZONTAL:RECORDLENGTH 100000</code> specifies that 100000 data points will be acquired for each record.<br><br><code>HORIZONTAL:RECORDLENGTH?</code> might return <code>:HORIZONTAL:RECORDlength 100000</code> indicating that the horizontal record length is equal to 100000 data points. |

## HORIZONTAL:RESOLUTION

Sets or returns the horizontal record length of acquired waveforms. The sample rate is automatically adjusted at the same time to maintain a constant time per division. The query form of this command returns the current horizontal record length.

|                  |   |
|------------------|---|
| <b>Group</b>     | Horizontal  |
| <b>Syntax</b>    | <code>HORIZONTAL:RESOLUTION &lt;NR1&gt;</code><br><code>HORIZONTAL:RESOLUTION?</code> |
| <b>Arguments</b> | <NR1> represents the supported values for horizontal record lengths.                  |

## HORIZONTAL:SAMPLERate

Returns the current horizontal sample rate.

|               |                                     |
|---------------|-------------------------------------|
| <b>Group</b>  | Horizontal                          |
| <b>Syntax</b> | <code>HORIZONTAL:SAMPLERate?</code> |

## HORIZONTAL:SCALE

Sets or returns the time base horizontal scale.

|                  |  |
|------------------|--|
| <b>Group</b>     | Horizontal   |
| <b>Syntax</b>    | HORizontal:SCALE <NR3><br>HORizontal:SCALE?  |
| <b>Arguments</b> | <NR3> specifies the range from 2 ns to 100 s, depending on the oscilloscope model.   |
| <b>Examples</b>  | HORIZONTAL:SCALE 2E-6 sets the main scale to 2µs per division.<br><br>HORIZONTAL:SCALE? might return :HORIZONTAL:MAIN:SCALE 2.0000E-06 indicating that the main scale is currently set to 2 µs per division. |

## HORizontal:TRIGger:POSition

Sets the horizontal position when delay mode is OFF. It is similar to [HORizontal:POSition](#).

|                 |  |
|-----------------|--|
| <b>Group</b>    | Horizontal   |
| <b>Syntax</b>   | HORizontal:TRIGger:POSition<br>HORizontal:TRIGger:POSition?  |
| <b>Examples</b> | HORIZONTAL:TRIGGER:POSITION? might return<br>HORIZONTAL:TRIGGER:POSITION 50 indicating that the horizontal trigger position is set to 50%. |

## ID? (Query Only)

Returns identifying information about the oscilloscope and related firmware.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Miscellaneous  |
| <b>Syntax</b>           | ID?  |
| <b>Related Commands</b> | <a href="#">*IDN?</a>  |
| <b>Examples</b>         | ID? might return TEK/DPO2024,CF:91.1CT,FV:v1.0000. This indicates the oscilloscope model number, configured format, and firmware version number. |

## \*IDN? (Query Only)

Returns the oscilloscope identification code.

**Group** Miscellaneous

**Syntax** \*IDN?

**Related Commands** [ID?](#)

**Examples** \*IDN? might return :TEKTRONIX,DPO2024,SN123456789,CF:91.1CT  
FV:v1.00000 indicating that the oscilloscope model number, serial number,  
configured number, and firmware version number.

## LANGUage

Sets or returns the user interface display language. This command only affects the oscilloscope displayed language. Remote commands and their responses are always in English.

**Group** Miscellaneous

**Syntax** LANGUage  
{ENGLISH|FRENCH|GERMAN|ITALIAN|SPANISH|PORTUGUESE|JAPANESE|  
KOREAN|RUSSIAN|SIMPLIFIEDCHINESE|TRADITIONALCHINESE}  
LANGUage?

**Examples** LANGUAGE? might return :LANGUAGE ENGLISH.

## LOCK

Enables or disables all front-panel buttons and knobs. There is no front panel equivalent.

When the front panel is locked, neither the [FPANEL:PRESS](#) nor the [FPANEL:TURN](#) commands will work. They will not generate an error event either. You can work around this by using the appropriate programmatic interface commands, instead of the front-panel commands. For example, to set the trigger level to 50%, you could use [TRIGGER:A SETLEVEL](#). To force a trigger, you could use [TRIGGER:FORCE](#).

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Miscellaneous   |
| <b>Syntax</b>           | LOCK {ALL NONE}<br>LOCK?  |
| <b>Related Commands</b> | <a href="#">UNLock</a>  |
| <b>Arguments</b>        | ALL disables all front-panel controls.<br><br>NONE enables all front-panel controls. This is equivalent to the UNLock ALL command.                  |
| <b>Examples</b>         | LOCK ALL locks the front-panel controls.<br><br>LOCK? might return :LOCK NONE indicating that the front-panel controls are enabled by this command. |

## \*LRN? (Query Only)

Returns the commands that list the oscilloscope settings except for configuration information for the calibration values, the [WFMInpre?](#) query, and the [WFMOutpre?](#) query. This query allows you to record or "learn" the current oscilloscope settings. You can use these commands to return the oscilloscope to the state it was in when you made the \*LRN? query. This command is identical to the [SET?](#) Command.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Miscellaneous   |
| <b>Syntax</b>           | *LRN?   |
| <b>Related Commands</b> | <a href="#">SET?</a>  |
| <b>Examples</b>         | *LRN? might return a long response, part of which could be as follows:<br>:ACQUIRE:STOPAFTER RUNSTOP;STATE 1;MODE SAMPLE;NUMENV<br>INFINITE;NUMAVG 16;SAMPLINGMODE RT;:HEADER 1;:LOCK<br>NONE;:LANGUAGE ENGLISH;:VERBOSE 1;:ALIAS:STATE<br>0;:DISPLAY:COLOR:PALETTE NORMAL;:DISPLAY:STYLE:DOTSONLY<br>0;:DISPLAY:PERSISTENCE 0.0000;CLOCK 1;GRATICULE<br>FULL;INTENSITY:WAVEFORM 30;GRATICULE 75;BACKLIGHT<br>HIGH;:HARDCOPY :INKSAVER OFF;LAYOUT LANDSCAPE;PREVIEW<br>0;:SAVE:IMAGE:FILEFORMAT BMP;:SAVE:WAVEFORM:FILEFORMAT |

```
INTERNAL;:SAVE:ASSIGN:TYPE SETUP;:TRIGGER:A:MODE
AUTO;TYPE EDGE;LE VEL 20.0000E-3;LEVEL:CH1
20.0000E-3;CH2 0.0000;CH3 0.0000;CH4 0.0000;:TRIGGER:A:
UPPERTHRESHOLD:CH1 1.4000;CH2 800.0000E-3;CH3
800.0000E-3;CH4 800.0000E-3;:TRIGG ER:A:LOWERTHRESHOLD:CH1
20.0000E-3;CH2 0.0000;CH3 0.0000;CH4 0.0000;:TRIGGER:A:H
OLDOFF:TIME 20.0000E-9;:TRIGGER:A:EDGE:SOURCE
CH1;COUPLING DC;SLOPE RISE;:TRIGGER:A:LOGIC:CLASS
SETHOLD;FUNCTION AND;THRESHOLD:CH1 20.0000E-3;CH2
0.0000;CH3 0.0 000;CH4 0.0000;:TRIGGER:A:LOGIC:INPUT:CH1
X;CH2 X;CH3 X;CH4 X;CLOCK:SOURCE NONE; EDGE
RISE;:TRIGGER:A:LOGIC:PATTERN:INPUT:CH1 X;CH2
X;CH3 X;CH4 X;:TRIGGER:A:LOGIC:PATTERN:WHEN
TRUE;WHEN:LESSLIMIT 4.0000E-9;LIMIT
4.0000E-9;:TRIGGER:A:SETHOLD:CLOCK:SOURCE CH1;EDGE
RISE;THRESHOLD 20.0000E-3;:TRIGGER:A:SETHOLD:DATA:SOURCE
CH2;THRESHOLD 0.0000;:TRIGGER:A:SETHOLD:HOLDTIME
4.0000E-9;SETTIME 4.0000E-9; :TRIGGER:A:PULSE:CLASS
TRANSITION;:TRIGGER:A:PULSEWIDTH:SOURCE
CH1;POLARITY POSITIVE;WHEN LESSTHAN;WIDTH
4.0000E-9;:TRIGGER:A:RUNT:SOURCE CH1;POLARITY POSITIVE; WHEN
OCCURS;WIDTH 4.0000E-9;:TRIGGER:A:TRANSITION:SOURCE CH1
...
```

## MARK

Moves to the next or previous reference mark on the waveform. Returns the current mark position.

**Group** Mark

**Syntax** MARK {NEXT|PREVIOUS}  
MARK?

**Arguments** NEXT moves to the next reference mark on the right.  
PREVIOUS moves to the next reference mark on the left.

## MARK:CREATE (No Query Form)

Creates a mark on a specified waveform or all waveforms in a column.

**Group** Mark

**Syntax** MARK:CREATE {CH<x>|MATH|B<x>|REF<x>|COLUMN|DIGITa1}

**Arguments** CH<x> creates the mark on a channel waveform, where <x> is the channel number.  
 MATH creates the mark on the math waveform.  
 B<x> creates the mark on a bus waveform, where <x> is the bus number.  
 REF<x> creates the mark on a reference waveform, where <x> is the reference waveform number.  
 COLUMN creates marks on all waveforms in the current zoom pixel column.  
 DIGITa1 creates marks on a digital waveform. (An error will result if no digital channel is turned on.)

## MARK:DELEte (No Query Form)

Deletes a mark on a particular waveform, all waveforms in a column, the selected mark, or all marks.

**Group** Mark

**Syntax** MARK:DELEte {CH<x>|MATH|B<x>|REF<x>|COLUMN|DIGITa1}

**Arguments** CH<x> deletes the mark on a channel waveform, where <x> is the channel number.  
 MATH deletes the mark on the math waveform.  
 B<x> deletes the mark on a bus waveform, where <x> is the bus number.  
 REF<x> deletes the mark on a reference waveform, where <x> is the reference waveform number.  
 COLUMN deletes marks on all waveforms in the current zoom pixel column.  
 DIGITa1 deletes all marks on all digital channels.

## MARK:FREE? (Query Only)

Returns how many marks are available for use.

**Group** Mark

**Syntax** MARK:FREE?

## MARK:SElected:END? (Query Only)

Returns the end of the selected mark, 0 to 100% of the waveform.

**Group** Mark

**Syntax** MARK:SElected:END?

## MARK:SElected:FOCUS? (Query Only)

Returns the focus of the selected mark, 0 to 100% of the waveform.

**Group** Mark

**Syntax** MARK:SElected:FOCUS?

## MARK:SElected:MARKSINCOLUMN? (Query Only)

Returns the number of marks in the current zoom pixel column.

**Group** Mark

**Syntax** MARK:SElected:MARKSINCOLUMN?

## MARK:SElected:OWNER? (Query Only)

Returns the owner of the selected mark.

**Group** Mark

**Syntax** MARK:SElected:OWNER?

**Returns** <QString> is the owner of the mark.

**Examples** MARK:SELECTED:OWNER? might return: USER, SEARCH1

## MARK:SElected:SOURCE? (Query Only)

Returns the source waveform for the selected mark.

**Group** Mark

**Syntax** MARK:SElected:SOURCE?

## MARK:SElected:START? (Query Only)

Returns the starting point of the selected mark, 0 to 100% of the waveform.

**Group** Mark

**Syntax** MARK:SElected:START?

## MARK:SElected:STATE? (Query Only)

Returns the on or off state of the selected mark. The selected mark is at or near the center of the screen. If you press the front-panel Set/Clear button, this mark will disappear.

**Group** Mark

**Syntax** MARK:SElected:STATE?

## MARK:SElected:ZOOM:POSITION? (Query Only)

Returns the position of the selected mark, 0 to 100% of the zoom overview window.

**Group** Mark

**Syntax** MARK:SElected:ZOOM:POSITION?



## MARK:TOTal? (Query Only)

Returns how many marks are currently in use.

**Group** Mark

**Syntax** MARK:TOTal?

## {MATH|MATH1}:LABel

Sets or queries the waveform label for the math waveform.

**Group** Math

**Syntax** {MATH|MATH1}:LABel <QString>  
{MATH|MATH1}:LABel?

**Arguments** <QString> is the quoted string used as the label for the math waveform.

**Examples** MATH:LABEL "Output" sets the label for the math waveform to Output.  
MATH:LABEL? might return MATH:LABEL "Sum of channel 1 and channel 2" indicating that the current label for the math waveform.

## MATH[1]? (Query Only)

Returns the definition of the math waveform. The returned data depends on the setting of the [MATH\[1\]:TYPe](#) command.

**Group** Math

**Syntax** MATH[1]?

**Related Commands** [MATH\[1\]:TYPe](#)

**Examples** MATH? or MATH1? might return :MATH:TYPE DUAL;DEFINE  
"CH1+CH2";VERTICAL:SCALE 100.0000E-3;POSITION 0.0000;UNITS

```
"V";:MATH:HORIZONTAL:SCALE 4.0000E-6;POSITION 50.0000;UNITS
"s";:MATH:SPECTRAL:MAG DB;WINDOW HANNING;:MATH:LABEL ""
```

## MATH[1]:DEFine

Sets or returns the current math function as a text string.

Dual math is defined if the string is of the form `<wfm> <operation> <wfm>`, where the `<wfm>`s are any combination of live channels or reference waveforms, `<operation>` is any of `+`, `-`, or `*`, and the `MATH[1]:TYPE` is `DUAL`.

FFT math is defined if the string is in the form `FFT(<wfm>)`, where `<wfm>` is any live channel or reference waveform, and the `MATH[1]:TYPE` is `FFT`.

On the front panel, the Dual Wfm Math, and FFT menus contain controls that allow building equivalent math expressions to those described above.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Math  |
| <b>Syntax</b>           | <code>MATH[1]:DEFine &lt;QString&gt;</code><br><code>MATH[1]:DEFine?</code>                                   |
| <b>Related Commands</b> | <a href="#">MATH[1]:TYPE</a>  |
| <b>Arguments</b>        | <code>&lt;QString&gt;</code> quoted string argument is the mathematical expression that defines the waveform. |

**Table 2-39: Advanced Math expression elements**

| Expression                           | Description  |
|--------------------------------------|--|
| CH1-CH4, REF1, REF2                  | Specifies a waveform data source.  |
| FFT(                                 | Executes a Fast Fourier Transform. The FFT operator must be the first (left-most) operator in an expression. All these operations must end with a right parenthesis. |
| AMPlitude( , AREa( , BURst( , CARea( | Executes the selected measurement operation on the waveform (active or reference) that follows. All these operations must end with a right parenthesis.              |
| + , - , *                            | Executes an addition, subtraction, or multiplication operation on the following expression.  |
| < , > , <= , >= , == , != ,    , &&  | Executes relational and logical operations.  |

**Examples**    `MATH1:DEFINE "CH1+CH2"` adds the Ch 1 waveform and Ch 2 waveform, storing the results in Math 1.

`MATH:DEFINE?` might return `:MATH1:DEFINE "CH2*REF2"` as the expression that defines Math 1.

## MATH[1]:HORizontal:POSition

Sets or returns the math horizontal display position for FFT or (non-live), live math reference waveforms.

**Group**    Math

**Syntax**    `MATH[1]:HORizontal:POSition <NR3>`  
`MATH[1]:HORizontal:POSition?`

**Arguments**    `<NR3>` is the % of the math waveform that precedes center screen. It can vary from 0.0 to 100.0.

**Examples**    `MATH:HORIZONTAL:POSITION 10` sets the horizontal position to 10% pretrigger

## MATH[1]:HORizontal:SCALE

Sets or returns the math horizontal display scale for FFT or (non-live), live math waveforms. The horizontal scale of a dual math waveform with a channel source waveform is set through the `HORizontal:SCALE` command.

**Group**    Math

**Syntax**    `MATH[1]:HORizontal:SCALE <NR3>`  
`MATH[1]:HORizontal:SCALE?`

**Arguments**    `<NR3>` is the math horizontal scale in seconds.

**Examples**    `MATH:HORIZONTAL:SCALE?` might return `MATH:HORIZONTAL:SCALE 2.0E-4` indicating that the math horizontal scale is 200  $\mu$

## MATH[1]:HORizontal:UNIts

Returns the math waveform horizontal measurement unit value.

|                 |  |
|-----------------|--|
| <b>Group</b>    | Math   |
| <b>Syntax</b>   | MATH[1]:HORizontal:UNIts?  |
| <b>Examples</b> | MATH:HORIZONTAL:UNITS? might return MATH:HORIZONTAL:UNITS "?" indicating that the math horizontal unit label for unknown values is the default question mark unit. |

## MATH[1]:SPECTral:MAG

Sets or returns the units of the Spectral Magnification function in the math string.

|                  |   |
|------------------|---|
| <b>Group</b>     | Math  |
| <b>Syntax</b>    | MATH[1]:SPECTral:MAG {LINEAR DB}<br>MATH[1]:SPECTral:MAG?   |
| <b>Arguments</b> | LINEAR sets the SpectralMag units to linear.<br>DB sets the SpectralMag units to decibels.  |
| <b>Examples</b>  | MATH1:SPECTRAL:MAG DB sets the SpectralMag units for Math1 to decibels.<br>MATH1:SPECTRAL:MAG? might return :MATH1:SPECTRAL:MAG DB indicating that the SpectralMag units for Math1 are set to decibels. |

## MATH[1]:SPECTral:WINDow

Sets or returns the window function for the spectral analyzer input data for the specified math waveform. A spectral window determines what the filter shape of the spectral analyzer will be in the frequency domain. It can be described by a mathematical function that is multiplied point-by-point times the input data to the spectral analyzer.

|              |      |
|--------------|------|
| <b>Group</b> | Math |
|--------------|------|

|                  |  |
|------------------|--|
| <b>Syntax</b>    | MATH[1]:SPECTral:WINDow<br>{RECTangular HAMming HANNing BLAckmanharris}<br>MATH[1]:SPECTral:WINDow?  |
| <b>Arguments</b> | RECTangular window function is equivalent to multiplying all gate data by one.<br>HAMming window function is based on a cosine series.<br>HANNing window function is based on a cosine series.<br>BLAckmanharris window function is based on a cosine series.                      |
| <b>Examples</b>  | MATH1:SPECTRAL:WINDOW HANNING applies a Hanning window to the spectral analyzer input data.<br><br>MATH1:SPECTRAL:WINDOW? might return :MATH1:SPECTRAL:WINDOW HAMMING indicating that the window function used to multiply the spectral analyzer input data is the Hamming window. |

## MATH[1]:TYPE

Sets or returns the math waveform mode type.

|                  |   |
|------------------|---|
| <b>Group</b>     | Math  |
| <b>Syntax</b>    | MATH[1]:TYPE {DUAL FFT}<br>MATH[1]:TYPE?  |
| <b>Arguments</b> | DUAL sets the math waveform mode to dual waveform math.<br>FFT sets the math waveform mode to FFT math.   |
| <b>Examples</b>  | MATH:TYPE FFT sets the math waveform mode to FFT.<br><br>MATH:TYPE FFT;:MATH:DEFINE "FFT( CH1 )" sets the math type to FFT and displays an FFT waveform of the channel 1 waveform, using the current FFT scale and window settings. |

## MATH[1]:VERTical:POSITION

Sets or returns the vertical position of the currently selected math type.

|              |      |
|--------------|------|
| <b>Group</b> | Math |
|--------------|------|

**Syntax** MATH[1]:VERTical:POSition <NR3>  
MATH[1]:VERTical:POSition?

**Related Commands** CH<x>:POSition, REF<x>:VERTical:POSition

**Arguments** <NR3> is the desired position in divisions from the center graticule.

**Examples** MATH1:VERTICAL:POSITION 1.3E+00 positions the Math 1 input signal 1.3 divisions higher than a position of 0.  
MATH1:VERTICAL:POSITION? might return :MATH1:VERTICAL:POSITION -1.3000E+00 indicating that the current position of Math 1 is 1.3 divisions below the center graticule.

## MATH[1]:VERTical:SCAle

Sets or returns the vertical scale of the currently selected math type.

**Group** Math

**Syntax** MATH[1]:VERTical:SCAle <NR3>  
MATH[1]:VERTical:SCAle?

**Related Commands** CH<x>:SCAle, REF<x>:VERTical:SCAle

**Arguments** <NR3> is the scale-per-division in the current math vertical units. The range is from 1.0E-09 through 5.0.

**Examples** MATH1:VERTICAL:SCALE 100E-03 sets the Math scale to 100 mV per division.  
MATH:VERTICAL:SCALE? might return :MATH:VERTICAL:SCALE 1.0000E+00 indicating that the current scale setting of Math is 1 V per division.

## MATH[1]:VERTical:UNIts? (Query Only)

Returns the math waveform vertical measurement unit value.

**Group** Math

**Syntax** MATH[1]:VERTICAL:UNITS?

**Examples** MATH:VERTICAL:UNITS? might return MATH:VERTICAL:UNITS "V" indicating that the math vertical unit label is volts.

## MEASUREMENT? (Query Only)

Returns all measurement parameters.

**Group** Measurement

**Syntax** MEASUREMENT?

**Examples** MEASUREMENT? might return :MEASUREMENT:IMMED:DELAY:DIRECTION FORWARDS;EDGE1 RISE;EDGE2 RISE;;MEASUREMENT:IMMED:TYPE PERIOD;UNITS "s";SOURCE1 CH1;SOURCE2 CH2;;MEASUREMENT:MEAS1:DELAY:DIRECTION FORWARDS;EDGE1 RISE;EDGE2 RISE;;MEASUREMENT:MEAS1:STATE 1;TYPE FREQUENCY;UNITS "Hz";SOURCE1 CH1;SOURCE2 CH2;COUNT 0;MAXIMUM 0.0000;MEAN 0.0000;MINIMUM 0.0000;STDDEV 0.0000;;MEASUREMENT:MEAS2:DELAY:DIRECTION FORWARDS;EDGE1 RISE;EDGE2 RISE;;MEASUREMENT:MEAS2:STATE 1;TYPE PERIOD;UNITS "s";SOURCE1 CH1;SOURCE2 CH2;COUNT 0;MAXIMUM 0.0000;MEAN 0.0000;MINIMUM 0.0000;STDDEV 0.0000;;MEASUREMENT:MEAS3:DELAY:DIRECTION FORWARDS;EDGE1 RISE;EDGE2 RISE;;MEASUREMENT:MEAS3:STATE 1;TYPE PK2PK;UNITS "V";SOURCE1 CH1;SOURCE2 CH2;COUNT 0;MAXIMUM 0.0000;MEAN 0.0000;MINIMUM 0.0000;STDDEV 0.0000;;MEASUREMENT:MEAS4:DELAY:DIRECTION FORWARDS;EDGE1 RISE;EDGE2 RISE;;MEASUREMENT:MEAS4:STATE 0;TYPE PERIOD;UNITS "s";SOURCE1 CH1;SOURCE2 CH2;COUNT 0;MAXIMUM 0.0000;MEAN 0.0000;MINIMUM 0.0000;STDDEV 0.0000;;MEASUREMENT:METHOD AUTO;REFLEVEL:METHOD PERCENT;ABSOLUTE:HIGH 0.0000;LOW 0.0000;MID1 0.0000;MID2 0.0000;;MEASUREMENT:REFLEVEL:PERCENT:HIGH 90.0000;LOW 10.0000;MID1 50.0000;MID2 50.0000;;MEASUREMENT:INDICATORS:STATE OFF;NUMHORZ 0;NUMVERT 0;HORZ1 99.0000E+36;HORZ2 99.0000E+36;HORZ3 99.0000E+36;HORZ4 99.0000E+36;VERT1 99.0000E+36;VERT2 99.0000E+36;VERT3 99.0000E+36;VERT4 99.0000E+36;;MEASUREMENT:STATISTICS:MODE OFF;WEIGHTING 32;;MEASUREMENT:GATING SCREEN.

## MEASUrement:CLEARSnapshot (No Query Form)

Removes the measurement snapshot display.

**Group** Measurement

**Syntax** MEASUrement:CLEARSnapshot

**Related Commands** [CLEARMenu](#)

## MEASUrement:GATing

Specifies or returns the measurement gating setting.

**Group** Measurement

**Syntax** MEASUrement:GATing {OFF|SCREen|CURSor}  
MEASUrement:GATing?

**Arguments** OFF turns off measurement gating (full record).  
SCREen turns on gating, using the left and right edges of the screen.  
CURSor limits measurements to the portion of the waveform between the vertical bar cursors, even if they are off screen.

**Examples** MEASUREMENT:GATING CURSOR turns on measurement gating using the cursors as limits.  
MEASUREMENT:GATING? might return :MEASUREMENT:GATING CURSOR indicating that measurements are limited to the portion of the waveform between the vertical bar cursors.

## MEASUrement:IMMed? (Query Only)

Returns all immediate measurement setup parameters.

**Group** Measurement

**Syntax** MEASUrement:IMMed?



**Examples** MEASUREMENT:IMMED? might return  
 :MEASUREMENT:IMMED:DELAY:DIRECTION FORWARDS;EDGE1 RISE;EDGE2  
 RISE; :MEASUREMENT:IMMED:TYPE PERIOD;UNITS "s";SOURCE1  
 CH1;SOURCE2 CH2

## MEASUREMENT:IMMED:DELAY? (Query Only)

Returns information about the immediate delay measurement. This command is equivalent to viewing the delay measurement settings on the measurement readout.

**Group** Measurement

**Syntax** MEASUREMENT:IMMED:DELAY?

**Examples** MEASUREMENT:IMMED:DELAY? might return  
 :MEASUREMENT:IMMED:DELAY:DIRECTION FORWARDS; EDGE1  
 RISE;EDGE2 RISE

## MEASUREMENT:IMMED:DELAY:DIRection

Sets or returns the starting point and direction that determines the delay "to" edge when taking an immediate delay measurement.

---

**NOTE.** Use the [MEASUREMENT:IMMED:SOURCE2](#) command to specify the delay "to" waveform.

---

**Group** Measurement

**Syntax** MEASUREMENT:IMMED:DELAY:DIRection {BACKWARDS|FORWARDS}  
 MEASUREMENT:IMMED:DELAY:DIRection?

**Related Commands** [MEASUREMENT:IMMED:SOURCE2](#)

**Arguments** BACKWARDS starts the search at the end of the waveform and looks for the last rising or falling edge in the waveform.

FORWARDS starts the search at the beginning of the waveform and looks for the first rising or falling edge in the waveform.

**Examples**      `MEASUREMENT:IMMED:DELAY:DIRECTION FORWARDS` starts searching from the beginning of the waveform record and looks for the first rising or falling edge.

`MEASUREMENT:IMMED:DELAY:DIRECTION?` might return  
`:MEASUREMENT:IMMED:DELAY:DIRECTION BACKWARDS` indicating that searching begins at the end of the waveform record and looks for the last rising or falling edge.

## MEASUREMENT:IMMED:DELAY:EDGE<x>

Sets or returns the slope of the edge the oscilloscope uses for the delay "from" or "to" waveform when taking an immediate delay measurement.

**Group**      Measurement

**Syntax**      `MEASUREMENT:IMMED:DELAY:EDGE<x> {FALL|RIS}`  
`MEASUREMENT:IMMED:DELAY:EDGE<x>?`

**Related Commands**      [MEASUREMENT:IMMED:SOURCE1](#)  
[MEASUREMENT:IMMED:SOURCE2](#)

**Arguments**      <x> specifies which waveform to use, where <x> = 1 is the "from" waveform, and <x> = 2 is the "to" waveform.

FALL specifies the falling edge.

RIS specifies the rising edge.

**Examples**      `MEASUREMENT:IMMED:DELAY:EDGE1 RISE` specifies that the "from" waveform rising edge be used for the immediate delay measurement.

`MEASUREMENT:IMMED:DELAY:EDGE1?` returns either RISE or FALL.

## MEASUREMENT:IMMED:SOURCE1

Sets or returns the source for all single source immediate measurements and specifies the source to measure "from" when taking an immediate delay measurement or phase measurement.

Digital channels (D<x>) are available as a measurement source for time, edge and pulse measurements such as Period, Frequency, Pos Width, Neg Width, Pos Duty Cycle, Neg Duty Cycle, Pos/Neg Edges and Pos/Neg Pulses, Delay and Phase.

**NOTE.** *If you do not specify a numerical suffix, the source is assumed to be SOURCE 1.*

**Group** Measurement

**Syntax** MEASUREMENT:IMMED:SOURCE1 {CH<x>|MATH<y>|REF<x>}  
MEASUREMENT:IMMED:SOURCE1?

**Related Commands** [MEASUREMENT:IMMED:SOURCE2](#)

**Arguments** CH<x> is an input channel waveform. The x variable can be expressed as an integer, where x is the channel number.

MATH<y> is a math waveform. The y variable can be expressed as an integer of 1.

REF<x> is a reference waveform. The x variable can be expressed as an integer, where x is the reference channel number.

**Examples** MEASUREMENT:IMMED:SOURCE1 MATH1  
specifies Math1 as the immediate measurement source.

MEASUREMENT:IMMED:SOURCE1? might return  
:MEASUREMENT:IMMED:SOURCE1 CH3 indicating that channel 3 is the immediate measurement source.

## MEASUREMENT:IMMED:SOURCE2

Sets or returns the source to measure "to" for phase or delay immediate measurements.

**NOTE.** *Source2 measurements only apply to phase and delay measurement types, which require both a target (Source1) and reference (Source2) source.*

**Group** Measurement

**Syntax** MEASUREMENT:IMMED:SOURCE2 {CH<x>|MATH<y>|REF<x>}  
MEASUREMENT:IMMED:SOURCE2?

**Related Commands** [MEASUREMENT:IMMED:SOURCE1](#)

**Arguments** CH<x> is an input channel waveform, where x is the channel number.  
 MATH<y> is a math waveform. The y variable can be expressed as an integer of 1.  
 REF<X> is a reference waveform, where x is the reference channel number.

**Examples** MEASUREMENT:IMMED:SOURCE2 REF2 sets the waveform in reference memory location 2 as the delay "to" source when making delay measurements.  
 MEASUREMENT:IMMED:SOURCE2? might return  
 :MEASUREMENT:IMMED:SOURCE2 MATH1 indicating that Math1 is the immediate measurement source.

## MEASUREMENT:IMMED:SOURCE<x>

For SOURCE1: Sets or returns the source for all single channel measurements. For delay or phase measurements, sets or returns the waveform to measure "from".

For SOURCE2: Sets or returns the waveform to measure "to" when taking a delay measurement or phase measurement.

**Group** Measurement

**Syntax** MEASUREMENT:IMMED:SOURCE<x> {CH1|CH2|CH3|CH4|MATH}  
 MEASUREMENT:IMMED:SOURCE<x>?

**Arguments** CH1-CH4 or MATH is the source waveform.

## MEASUREMENT:IMMED:TYPE

Sets or returns the immediate measurement type.

**Group** Measurement

**Syntax** MEASUREMENT:IMMED:TYPE  
 {AMPLITUDE|AREA|BURST|CAREA|CMean|CRMS|DELAY|FALL|FREQUENCY  
 |HIGH|LOW|MAXIMUM|MEAN|MINIMUM|NDUTY|NEDGECount|NOVERSHOOT  
 |NPULSECount|NWIDTH|PEDGECount|PDUTY  
 |PERIOD|PHASE|PK2Pk|POVERSHOOT|PPULSECount|PWIDTH|RISE|RMS}  
 MEASUREMENT:IMMED:TYPE?

**Arguments** **AMPLi tude** measures the amplitude of the selected waveform. In other words, it measures the high value less the low value measured over the entire waveform or gated region. This measurement is available only on DPO models.

$$\textit{Amplitude} = \textit{High} - \textit{Low}$$

**AREa** measures the voltage over time. The area is over the entire waveform or gated region and is measured in volt-seconds. The area measured above the ground is positive, while the area below ground is negative. This measurement is available only on DPO models.

**BURSt** measures the duration of a burst. The measurement is made over the entire waveform or gated region.

**CAREa** (cycle area) measures the voltage over time. In other words, it measures, in volt-seconds, the area over the first cycle in the waveform or the first cycle in the gated region. The area measured above the common reference point is positive, while the area below the common reference point is negative. This measurement is available on DPO and MSO models.

**CMEan** (cycle mean) measures the arithmetic mean over the first cycle in the waveform or the first cycle in the gated region. This measurement is available only on DPO and MSO models.

**CRMS** (cycle rms) measures the true Root Mean Square voltage over the first cycle in the waveform or the first cycle in the gated region. This measurement is available only on DPO and MSO models.

**DELay** measures the time between the middle reference (default = 50%) amplitude point of the source waveform and the destination waveform.

**FALL** measures the time taken for the falling edge of the first pulse in the waveform or gated region to fall from a high reference value (default is 90%) to a low reference value (default is 10%). This measurement is available only on DPO models.

**FREQUency** measures the first cycle in the waveform or gated region. Frequency is the reciprocal of the period and is measured in hertz (Hz), where 1 Hz = 1 cycle per second.

**HIGH** measures the High reference (100% level, sometimes called Topline) of a waveform. This measurement is available only on DPO models.

**LOW** measures the Low reference (0% level, sometimes called Baseline) of a waveform. This measurement is available only on DPO models.

**MAXimum** finds the maximum amplitude. This value is the most positive peak voltage found. It is measured over the entire waveform or gated region. This measurement is available only on DPO models.

**MEAN** amplitude measurement finds the arithmetic mean over the entire waveform or gated region. This measurement is available only on DPO models.

**MINIMUM** finds the minimum amplitude. This value is typically the most negative peak voltage. It is measured over the entire waveform or gated region. This measurement is available only on DPO models.

**NDUTY** (negative duty cycle) is the ratio of the negative pulse width to the signal period, expressed as a percentage. The duty cycle is measured on the first cycle in the waveform or gated region.

$$\text{Negative Duty Cycle} = ((\text{Negative Width}) / \text{Period}) \times 100\%$$

**NEDGECount** is the count of falling edges.

**NOVERSHOOT** (negative overshoot) finds the negative overshoot value over the entire waveform or gated region. This measurement is available only on DPO models.

$$\text{Negative Overshoot} = ((\text{Low} - \text{Minimum}) / \text{Amplitude}) \times 100\%$$

**NPULSECount** is the count of negative pulses.

**NWIDTH** (negative width) measurement is the distance (time) between the middle reference (default = 50%) amplitude points of a negative pulse. The measurement is made on the first pulse in the waveform or gated region.

**PDUTY** (positive duty cycle) is the ratio of the positive pulse width to the signal period, expressed as a percentage. It is measured on the first cycle in the waveform or gated region.

$$\text{Positive Duty Cycle} = ((\text{Positive Width} / \text{Period}) \times 100\%$$

**PEDGECount** is the count of rising edges.

**PERIOD** is the time required to complete the first cycle in a waveform or gated region. Period is the reciprocal of frequency and is measured in seconds.

**PHASE** measures the phase difference (amount of time a waveform leads or lags the reference waveform) between two waveforms. The measurement is made between the middle reference points of the two waveforms and is expressed in degrees, where 360° represents one waveform cycle.

**PK2Pk** (peak-to-peak) finds the absolute difference between the maximum and minimum amplitude in the entire waveform or gated region. This measurement is available only on DPO models.

**POVERSHOOT** is the positive overshoot value over the entire waveform or gated region. This measurement is available only on DPO models.

$$\text{Positive Overshoot} = ((\text{Maximum} - \text{High}) / \text{Amplitude}) \times 100\%$$

**PPULSECount** is the count of positive pulses.

**PWIDTH** (positive width) is the distance (time) between the middle reference (default = 50%) amplitude points of a positive pulse. The measurement is made on the first pulse in the waveform or gated region.

**RISe** timing measurement finds the rise time of the waveform. The rise time is the time it takes for the leading edge of the first pulse encountered to rise from a low reference value (default is 10%) to a high reference value (default is 90%). This measurement is available only on DPO models.

**RMS** amplitude measurement finds the true Root Mean Square voltage in the entire waveform or gated region. This measurement is available only on DPO models.

**Examples** `MEASUREMENT:IMMED:TYPE FREQUENCY` defines the immediate measurement to be a frequency measurement.

`MEASUREMENT:IMMED:TYPE?` might return `:MEASUREMENT:IMMED:TYPE RMS` indicating that the immediate measurement is the true Root Mean Square voltage.

## MEASUREMENT:IMMED:UNITS? (Query Only)

Returns the units of the immediate measurement:

VOLTS, VOLTS SQUARED, SEC, HERTZ, PERCENT, DIVS, SAMPLES, OHMS, AMPS, WATTS, MINUTES, DEGREES, UNKNOWN, AMPS SQUARED, HOURS, DAYS, DB, BYTES, INVERSE HERTZ, IRE, V OVER V, V OVER A, VOLTS WATTS, V OVER W, VOLTS DB, V OVER DB, A OVER V, A OVER A, AMPS WATTS, A OVER W, AMPS DB, A OVER DB, WATTS VOLTS, W OVER V, WATTS AMPS, W OVER A, WATTS SQUARED, W OVER W, WATTS DB, W OVER DB, DB VOLTS, DB OVER V, DB AMPS, DB OVER A, DB WATTS, DB OVER W, DB SQUARED, DB OVER DB, VOLTS SEC, AMPS SEC, WATTS SEC, V OVER S, A OVER S, W OVER S

**Group** Measurement

**Syntax** `MEASUREMENT:IMMED:UNITS?`

**Examples** `MEASUREMENT:IMMED:UNITS?` might return

`:MEASUREMENT:IMMED:UNITS "s"`

indicating that units for the immediate measurement are in seconds.

## MEASUREMENT:IMMED:VALUE? (Query Only)

Returns the value of the measurement specified by the [MEASUREMENT:IMMED:TYPE](#) command. The measurement is immediately taken on the source(s) specified by a [MEASUREMENT:IMMED:SOURCE1](#) command.

---

**NOTE.** A change to *HORizontal:MAIn:SCALE* or *CH<x>:SCALE* will not necessarily have taken affect if immediately followed by this command.

---

**Group** Measurement

**Syntax** MEASUrement:IMMed:VALue?

**Related Commands** [MEASUrement:IMMed:TYPe](#), [MEASUrement:IMMed:SOUrce1](#), [\\*ESR?](#), [ALLEv?](#)

**Examples** MEASUREMENT:IMMED:VALUE? might return :MEASUREMENT:IMMED:VALUE 9.9000E+37. If the measurement has an error or warning associated with it, then an item is added to the error queue. The error can be checked for with the [\\*ESR?](#) and [ALLEv?](#) commands.

## MEASUrement:INDICators? (Query Only)

Returns all measurement indicator parameters.

**Group** Measurement

**Syntax** MEASUrement:INDICators?

**Examples** MEASUREMENT:INDICATORS? might return  
 MEASUREMENT:INDICATORS:STATE MEAS1;NUMHORZ 0;NUMVERT 4;HORZ1  
 7.5E0;HORZ2 -3.40000095367E0;HORZ3 0.0E0;HORZ4 0.0E0;VERT1  
 -6.351123E-6;VERT2 -3.179753E-6;VERT3 -6.40943E-6;VERT4  
 -6.403E-6

## MEASUrement:INDICators:HORIZ<x>? (Query Only)

Returns the position of the specified horizontal measurement indicator <x>, where <x> can be 1, 2, 3, or 4.

**Group** Measurement

**Syntax** MEASUrement:INDICators:HORIZ<x>?



**Examples** MEASUREMENT:INDICATORS:HORIZ1? might return  
MEASUREMENT:INDICATORS:HORIZ1 -2.0E-3 indicating that horizontal  
indicator1 has a value of -2 mV.

## MEASUREMENT:INDICATORS:NUMHORZ? (Query Only)

Returns the number of horizontal measurement indicators currently being displayed.

**Group** Measurement

**Syntax** MEASUREMENT:INDICATORS:NUMHORZ?

**Examples** MEASUREMENT:INDICATORS:NUMHORZ? might return  
MEASUREMENT:INDICATORS:NUMHORZ 2 indicating that there are currently  
2 horizontal lines drawn on the graticule. The indicators show where the  
measurement specified by [MEASUREMENT:INDICATORS:STATE](#) is being  
performed.

## MEASUREMENT:INDICATORS:NUMVERT? (Query Only)

Returns the number of vertical measurement indicators currently being displayed.

**Group** Measurement

**Syntax** MEASUREMENT:INDICATORS:NUMVERT?

**Examples** MEASUREMENT:INDICATORS:NUMVERT? might return  
MEASUREMENT:INDICATORS:NUMVERT 2 indicating that there are currently 2  
vertical lines drawn on the graticule. The indicators show where the measurement  
specified by [MEASUREMENT:INDICATORS:STATE](#) is being performed.

## MEASUREMENT:INDICATORS:STATE

Sets or returns the state of visible measurement indicators.

**Group** Measurement

**Syntax** MEASUREMENT:INDICATORS:STATE {OFF|MEAS<x>}  
MEASUREMENT:INDICATORS:STATE?

**Arguments** OFF turns the visible measurement indicators off.  
MEAS<x> displays the visible measurement indicators for measurement <x>, where <x> can be 1, 2, 3, or 4.

---

**NOTE.** *There must be an active measurement before you can activate an indicator for a specified measurement.*

---

**Examples** MEASUREMENT:INDICATORS:STATE MEAS2 turns on the display of visible measurement indicators for measurement 2.  
MEASUREMENT:INDICATORS:STATE? might return  
MEASUREMENT:INDICATORS:STATE OFF indicating that no measurement indicators are active.

## MEASUREMENT:INDICATORS:VERT<x>? (Query Only)

Returns the value of the specified vertical measurement indicator <x> from the trigger point, where <x> can be 1, 2, 3, or 4. A negative value means that the indicator is positioned earlier in the waveform record than the trigger point.

**Group** Measurement

**Syntax** MEASUREMENT:INDICATORS:VERT<x>?

**Examples** MEASUREMENT:INDICATORS:VERT2? might return  
MEASUREMENT:INDICATORS:VERT2 -3.724507E-6 indicating that the second measurement indicator is positioned 3.72  $\mu$ s before the trigger point.

## MEASUREMENT:MEAS<x>? (Query Only)

Returns all measurement parameters for the specified active measurement <x>.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>?

## MEASUREMENT:MEAS<x>:COUNT? (Query Only)

Returns the number of values accumulated for this measurement since the last statistical reset. Values may be ignored if they generated an error. Measurements are specified by x, which ranges from 1 through 4.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:COUNT?

**Examples** MEASUREMENT:MEAS3:COUNT? might return :MEASUREMENT:MEAS3:COUNT 3247.

## MEASUREMENT:MEAS<x>:DELAY? (Query Only)

Returns the delay measurement parameters for the measurement specified by <x>, which ranges from 1 through 4.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:DELAY?

**Examples** MEASUREMENT:MEAS1? might return :MEASUREMENT:MEAS1:DELAY:DIRECTION FORWARDS;EDGE1 RISE;EDGE2 RISE.

## MEASUREMENT:MEAS<x>:DELAY:DIRection

Sets or returns the starting point and direction that determines the delay "to" edge when taking a delay measurement. Use the [MEASUREMENT:MEAS<x>:SOURCE2](#) command to specify the waveform.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:DELAY:DIRection {BACKWARDS|FORWARDS}  
MEASUREMENT:MEAS<x>:DELAY:DIRection?

**Related Commands** [MEASUREMENT:MEAS<x>:SOURCE2](#)

**Arguments**      **BACKWARDS** means the search starts at the end of the waveform and looks for the last rising or falling edge in the waveform. Use the [MEASUREMENT:MEAS<x>:DELAY:EDGE<x>](#) command to specify the slope of the edge.

**FORWARDS** means the search starts at the beginning of the waveform and looks for the first rising or falling edge in the waveform. Use the [MEASUREMENT:MEAS<x>:DELAY:EDGE<x>](#) command to specify the slope of the edge.

**Examples**      `MEASUREMENT:MEAS3:DELAY:DIRECTION BACKWARDS` starts searching from the end of the waveform record.

`MEASUREMENT:MEAS3:DELAY:DIRECTION?` might return  
`:MEASUREMENT:MEAS3:DELAY:DIRECTION BACKWARDS` indicating that the current search direction is backwards.

## MEASUREMENT:MEAS<x>:DELAY:EDGE<x>

Sets or returns the slope of the edge used for the delay "from" or "to" waveform when taking an immediate delay measurement. The waveform is specified by [MEASUREMENT:MEAS<x>:SOURCE\[1\]](#).

**Group**      Measurement

**Syntax**      `MEASUREMENT:MEAS<x>:DELAY:EDGE<x> {FALL|RISE}`  
`MEASUREMENT:MEAS<x>:DELAY:EDGE<x>?`

**Arguments**      <x> specifies which waveform to use, where <x> = 1 is the "from" waveform, and <x> = 2 is the "to" waveform.

**FALL** specifies the falling edge.

**RISE** specifies the rising edge.

**Examples**      `MEASUREMENT:MEAS1:DELAY:EDGE1 RISE` specifies that the "from" waveform rising edge be used for the immediate delay measurement.

`MEASUREMENT:MEAS1:DELAY:EDGE1?` returns either **RISE** or **FALL**.

## MEASUREMENT:MEAS<x>:MAXIMUM? (Query Only)

Returns the maximum value found for this measurement since the last statistical reset. Measurements are specified by x, which ranges from 1 through 4.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:MAXIMUM?

**Examples** MEASUREMENT:MEAS3:MAXIMUM? might return  
:MEASUREMENT:MEAS3:MAXIMUM 4.18E-9.

## MEASUREMENT:MEAS<x>:MEAN? (Query Only)

Returns the mean value accumulated for this measurement since the last statistical reset. Measurements are specified by x, which ranges from 1 through 4.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:MEAN?

**Examples** MEASUREMENT:MEAS1:MEAN? might return :MEASUREMENT:MEAS1:MEAN  
514.71E-09.

## MEASUREMENT:MEAS<x>:MINIMUM? (Query Only)

Returns the minimum value for this measurement since the last statistical reset. Measurements are specified by <x>, which ranges from 1 through 4.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:MINIMUM?

**Examples** MEASUREMENT:MEAS1:MINIMUM? might return  
:MEASUREMENT:MEAS1:MINIMUM 1.75E-09.

## MEASUREMENT:MEAS<x>:SOURCE[1]

Sets or returns the source for all single source measurements and specifies the source to measure "from" when taking a delay measurement or phase measurement. Measurements are specified by <x>, which ranges from 1 through 4.

|                  |   |
|------------------|---|
| <b>Group</b>     | Measurement   |
| <b>Syntax</b>    | MEASUREMENT:MEAS<x>:SOURCE[1] {CH<x> MATH<y> REF<x>}<br>MEASUREMENT:MEAS<x>:SOURCE[1]?  |
| <b>Arguments</b> | CH<x> is an input channel waveform, where x is the channel number.<br>MATH<y> is a math waveform, where y is 1.<br>REF<x> is a reference waveform, where x is the reference channel number.                       |
| <b>Examples</b>  | MEASUREMENT:MEAS2:SOURCE1 MATH1 specifies Math 1 as the measurement 2 source.<br>MEASUREMENT:MEAS1:SOURCE1? might return<br>:MEASUREMENT:MEAS1:SOURCE[1] MATH1 indicating that Math1 is the measurement 2 source. |

## MEASUREMENT:MEAS<x>:SOURCE2

Sets or returns the reference source to measure "to" when taking a delay measurement or phase measurement. Measurements are specified by <x>, which ranges from 1 through 4.

---

**NOTE.** *Source2 measurements only apply to phase and delay measurement types, which require both a target (Source1) and reference (Source2) source.*

---

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Measurement  |
| <b>Syntax</b>           | MEASUREMENT:MEAS<x>:SOURCE2 {CH<x> MATH<y> REF<x>}<br>MEASUREMENT:MEAS<x>:SOURCE2?   |
| <b>Related Commands</b> | <a href="#">MEASUREMENT:MEAS&lt;x&gt;:TYPE</a>   |
| <b>Arguments</b>        | CH<x> is an input channel waveform, where x is the channel number.<br>MATH<y> is the math waveform, which is always 1.<br>REF<x> is a reference waveform, where x is the reference channel number. |
| <b>Examples</b>         | MEASUREMENT:MEAS4:SOURCE2 CH1 specifies CH1 as the delay "to" source when making delay measurement.  |

MEASUREMENT:MEAS2:SOURCE2? might return  
:MEASUREMENT:MEAS2:SOURCE2 MATH1 indicating that Math 1 is the  
measurement 2 source.

## MEASUREMENT:MEAS<x>:SOURCE<x>

For SOURCE1: Sets or returns the source for all single channel measurements. For delay or phase measurements, sets or returns the waveform to measure "from".

For SOURCE2: Sets or returns the waveform to measure "to" when taking a delay measurement or phase measurement.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:SOURCE<x> {CH<x>|MATH|}  
MEASUREMENT:MEAS<x>:SOURCE<x>?

**Arguments** CH<x> is an input channel waveform, where x is the channel number.  
MATH is the math waveform.  
REF<x> is a reference waveform, where x is the reference channel number.

## MEASUREMENT:MEAS<x>:STATE

Sets or returns whether the specified measurement slot is computed and displayed. The measurement slot is specified by <x>, which ranges from 1 through 4.

For a measurement to display, you must have selected a source waveform and defined the measurement you want to take and display. You select the measurement using the [MEASUREMENT:MEAS<x>:SOURCE\[1\]](#) command. You define the measurement type using the [MEASUREMENT:MEAS<x>:TYPE](#) command.

**Group** Measurement

**Syntax** MEASUREMENT:MEAS<x>:STATE {OFF|ON|<NR1>}  
MEASUREMENT:MEAS<x>:STATE?

**Related Commands** [MEASUREMENT:MEAS<x>:SOURCE\[1\]](#), [MEASUREMENT:MEAS<x>:TYPE](#)

|                  |   |
|------------------|---|
| <b>Arguments</b> | <p>OFF disables calculation and display of the specified measurement slot.</p> <p>ON enables calculation and display of the specified measurement slot.</p> <p>&lt;NR1&gt; = 0 disables calculation and display of the specified measurement slot; any other value enables calculation and display of the specified measurement slot.</p> |
| <b>Examples</b>  | <p>MEASUREMENT:MEAS2:STATE ON computes and displays the measurement defined as measurement 2.</p> <p>MEASUREMENT:MEAS1:STATE? might return :MEASUREMENT:MEAS1:STATE 0 indicating that measurement defined for measurement slot 1 is disabled.</p>   |

## MEASUREMENT:MEAS<x>:STDdev? (Query Only)

Returns the standard deviation of values accumulated for this measurement since the last statistical reset. Measurements are specified by <x>, the measurement slots, from 1 through 4.

|                 |  |
|-----------------|--|
| <b>Group</b>    | Measurement  |
| <b>Syntax</b>   | MEASUREMENT:MEAS<x>:STDdev?  |
| <b>Examples</b> | MEASUREMENT:MEAS1:STDDEV? might return :MEASUREMENT:MEAS1:STDDEV 21.0E-12. |

## MEASUREMENT:MEAS<x>:TYPE

Sets or returns the measurement type defined for the specified measurement slot. The measurement slot is specified by <x>, which ranges from 1 through 4.

|               |  |
|---------------|--|
| <b>Group</b>  | Measurement  |
| <b>Syntax</b> | <p>MEASUREMENT:MEAS&lt;x&gt;:TYPE</p> <p>{AMP ititude AREa BURSt CAREa CMEan CRMs DELay FALL FREQUency HIGH LOW MAXimum MEAN MINImum NDuty NEDGECount NOVershoot NPULSECount NWIdth PDuty PEDGECount PERIOD PHase PK2Pk POVershoot PPULSECount PWidth RISe RMS}</p> <p>MEASUREMENT:MEAS&lt;x&gt;:TYPE?</p> |



**Arguments** **AMP**litude measures the amplitude of the selected waveform. In other words, it measures the high value less the low value measured over the entire waveform or gated region. This measurement is available only on DPO models.

$$\textit{Amplitude} = \textit{High} - \textit{Low}$$

**ARE**a measures the voltage over time. The area is over the entire waveform or gated region and is measured in volt-seconds. The area measured above the ground is positive, while the area below ground is negative. This measurement is available only on DPO models.

**BUR**st measures the duration of a burst. The measurement is made over the entire waveform or gated region.

**CARE**a (cycle area) measures the voltage over time. In other words, it measures, in volt-seconds, the area over the first cycle in the waveform or the first cycle in the gated region. The area measured above the common reference point is positive, while the area below the common reference point is negative. This measurement is available only on DPO models.

**CME**an (cycle mean) measures the arithmetic mean over the first cycle in the waveform or the first cycle in the gated region. This measurement is available only on DPO models.

**CRMS** (cycle rms) measures the true Root Mean Square voltage over the first cycle in the waveform or the first cycle in the gated region. This measurement is available only on DPO models.

**DEL**ay measures the time between the middle reference (default = 50%) amplitude point of the source waveform and the destination waveform. This measurement is available only on DPO models.

**FALL** measures the time taken for the falling edge of the first pulse in the waveform or gated region to fall from a high reference value (default is 90%) to a low reference value (default is 10%). This measurement is available only on DPO models.

**FRE**quency measures the first cycle in the waveform or gated region. Frequency is the reciprocal of the period and is measured in hertz (Hz), where 1 Hz = 1 cycle per second.

**HIGH** measures the High reference (100% level, sometimes called Topline) of a waveform. This measurement is available only on DPO models.

**LOW** measures the Low reference (0% level, sometimes called Baseline) of a waveform. This measurement is available only on DPO models.

**MAX**imum finds the maximum amplitude. This value is the most positive peak voltage found. It is measured over the entire waveform or gated region. This measurement is available only on DPO models.

**MEAN** amplitude measurement finds the arithmetic mean over the entire waveform or gated region. This measurement is available only on DPO models.

**MINIMUM** finds the minimum amplitude. This value is typically the most negative peak voltage. It is measured over the entire waveform or gated region. This measurement is available only on DPO models.

**NDUTY** (negative duty cycle) is the ratio of the negative pulse width to the signal period, expressed as a percentage. The duty cycle is measured on the first cycle in the waveform or gated region.

$$\text{Negative Duty Cycle} = ((\text{Negative Width}) / \text{Period}) \times 100\%$$

**NEDGECount** is the count of negative edges.

**NOVERSHOOT** (negative overshoot) finds the negative overshoot value over the entire waveform or gated region. This measurement is available only on DPO models.

$$\text{Negative Overshoot} = ((\text{Low} - \text{Minimum}) / \text{Amplitude}) \times 100\%$$

**NPULSECount** is the count of negative pulses.

**NWIDTH** (negative width) measurement is the distance (time) between the middle reference (default = 50%) amplitude points of a negative pulse. The measurement is made on the first pulse in the waveform or gated region.

**PDUTY** (positive duty cycle) is the ratio of the positive pulse width to the signal period, expressed as a percentage. It is measured on the first cycle in the waveform or gated region.

$$\text{Positive Duty Cycle} = ((\text{Positive Width}) / \text{Period}) \times 100\%$$

**PEDGECount** is the count of positive edges.

**PERIOD** is the time required to complete the first cycle in a waveform or gated region. Period is the reciprocal of frequency and is measured in seconds.

**PHASE** measures the phase difference (amount of time a waveform leads or lags the reference waveform) between two waveforms. The measurement is made between the middle reference points of the two waveforms and is expressed in degrees, where 360° represents one waveform cycle.

**PK2Pk** (peak-to-peak) finds the absolute difference between the maximum and minimum amplitude in the entire waveform or gated region. This measurement is available only on DPO models.

**POVERSHOOT** is the positive overshoot value over the entire waveform or gated region. This measurement is available only on DPO models.

$$\text{Positive Overshoot} = ((\text{Maximum} - \text{High}) / \text{Amplitude}) \times 100\%$$

**PPULSECount** is the count of positive pulses.

**PWIDTH** (positive width) is the distance (time) between the middle reference (default = 50%) amplitude points of a positive pulse. The measurement is made on the first pulse in the waveform or gated region.

**RISe** timing measurement finds the rise time of the waveform. The rise time is the time it takes for the leading edge of the first pulse encountered to rise from a low reference value (default is 10%) to a high reference value (default is 90%). This measurement is available only on DPO models.

**RMS** amplitude measurement finds the true Root Mean Square voltage in the entire waveform or gated region. This measurement is available only on DPO models.

**Examples** `MEASUREMENT:MEAS2:TYPE FREQUENCY` defines measurement 2 as a measurement of the frequency of a waveform.

`MEASUREMENT:MEAS1:TYPE?` might return `:MEASUREMENT:MEAS1:TYPE RMS` indicating that measurement 1 is defined to measure the RMS value of a waveform.

## MEASUREMENT:MEAS<x>:UNITS? (Query Only)

Returns the units associated with the specified measurement. The measurement slots are specified by <x>, which ranges from 1 through 4.

**Group** Measurement

**Syntax** `MEASUREMENT:MEAS<x>:UNITS?`

**Related Commands** [MEASUREMENT:MEAS<x>:TYPE](#)

**Examples** `MEASUREMENT:MEAS1:UNITS?` might return `:MEASUREMENT:MEAS1:UNITS %` indicating units for measurement 1 are set to percent.

## MEASUREMENT:MEAS<x>:VALUE? (Query Only)

Returns a calculate value for the measurement specified by <x>, which ranges from 1 through 4.

---

**NOTE.** *This is the same value as displayed on-screen. If measurement statistics are enabled, a new value is calculated with every waveform. In addition, this value is updated approximately every 1/3 second. If you are acquiring a long acquisition record, the oscilloscope may take longer to update.*

---

**Group** Measurement

**Syntax**      MEASUREMENT:MEAS<x>:VALUE?

**Related Commands**      [MEASUREMENT:MEAS<x>:UNITS?](#), [\\*ESR?](#), [ALLEV?](#)

**Examples**      MEASUREMENT:MEAS1:VALUE? might return :MEASUREMENT:MEAS1:VALUE 2.8740E-06. If the measurement has an error or warning associated with it, then an item is added to the error queue. The error can be checked for with the [\\*ESR?](#) and [ALLEV?](#) commands.

## MEASUREMENT:METHOD

Sets or returns the method used to calculate the 0% and 100% reference level.

**Group**      Measurement

**Syntax**      MEASUREMENT:METHOD {Auto|HISTogram|MINMax}  
MEASUREMENT:METHOD?

**Related Commands**      [MEASUREMENT:REFLevel:PERCent:HIGH](#), [MEASUREMENT:REFLevel:PERCent:LOW](#), [MEASUREMENT:REFLevel:PERCent:MID](#), [MEASUREMENT:REFLevel:PERCent:MID2](#)

**Arguments**      Auto selects the best method for each data set.

HISTogram sets the high and low waveform levels statistically using a histogram algorithm.

MINMax uses the highest and lowest values of the waveform record. This selection is best for examining waveforms with no large, flat portions of a common value, such as sine waves and triangle waves.

**Examples**      MEASUREMENT:METHOD? might return :MEASUREMENT:METHOD MINMAX indicating that the reference levels are set to MIN and MAX.

## MEASUREMENT:REFLevel? (Query Only)

Returns the current reference level parameters.

**Group**      Measurement

**Syntax** MEASUREMENT:REFLEVEL?

**Examples** MEASUREMENT:REFLEVEL? might return these reference level settings :MEASUREMENT:REFLEVEL:METHOD PERCENT;ABSOLUTE:HIGH 0.0000;LOW 0.0000;MID1 0.0000 ;MID2 0.0000;:MEASUREMENT:REFLEVEL:PERCENT:HIGH 90.0000;LOW 10.0000;MID1 50.0000 ;MID2 50.0000

## MEASUREMENT:REFLEVEL:ABSOLUTE:HIGH

Sets or returns the high reference level, and is the upper reference level when [MEASUREMENT:REFLEVEL:METHOD](#) is set to Absolute. This command affects the results of rise and fall measurements.

---

**NOTE.** *this command affects the associated reference level parameter for all MEASUREMENTS:IMMED and the four periodic measurements.*

---

**Group** Measurement

**Syntax** MEASUREMENT:REFLEVEL:ABSOLUTE:HIGH <NR3>  
MEASUREMENT:REFLEVEL:ABSOLUTE:HIGH?

**Related Commands** [MEASUREMENT:REFLEVEL:METHOD](#), [MEASUREMENT:IMMED:TYPE](#), [MEASUREMENT:MEAS<x>:TYPE](#)

**Arguments** <NR3> is the high reference level, in volts. The default is 0.0 V.

**Examples** MEASUREMENT:REFLEVEL:ABSOLUTE:HIGH 1.71 sets the high reference level to 1.71 V.

MEASUREMENT:REFLEVEL:ABSOLUTE:HIGH? might return :MEASUREMENT:REFLEVEL:ABSOLUTE:HIGH 1.7100E+00 indicating that the absolute high reference level is set to 1.71 V.

## MEASUREMENT:REFLEVEL:ABSOLUTE:LOW

Sets or returns the low reference level, and is the lower reference level when [MEASUREMENT:REFLEVEL:METHOD](#) is set to Absolute.

---

**NOTE.** *this command affects the associated reference level parameter for all MEASurements:IMMed and the four periodic measurements.*

---

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Measurement  |
| <b>Syntax</b>           | MEASUREMENT:REFLevel:ABSolute:LOW <NR3><br>MEASUREMENT:REFLevel:ABSolute:LOW?  |
| <b>Related Commands</b> | <a href="#">MEASUREMENT:REFLevel:METHod</a> , <a href="#">MEASUREMENT:IMMed:TYPe</a> ,<br><a href="#">MEASUREMENT:MEAS&lt;x&gt;:TYPe</a>   |
| <b>Arguments</b>        | <NR3> is the low reference level, in volts. The default is 0.0 V.  |
| <b>Examples</b>         | MEASUREMENT:REFLEVEL:ABSOLUTE:LOW 0.0 sets the low reference level to 0.0 V.<br><br>MEASUREMENT:REFLEVEL:ABSOLUTE:LOW? might return<br>:MEASUREMENT:REFLEVEL:ABSOLUTE:LOW 0.0000E+00 indicating that the absolute low reference level is set to 0.0 V. |

## MEASUREMENT:REFLevel:ABSolute:MID[1]

Sets or returns the mid reference level, and is the 50% reference level when [MEASUREMENT:REFLevel:METHod](#) is set to Absolute. This command affects the results of period, frequency, delay, and all cyclic measurements.

---

**NOTE.** *this command affects the associated reference level parameter for all MEASurements:IMMed and the four periodic measurements.*

---

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Measurement   |
| <b>Syntax</b>           | MEASUREMENT:REFLevel:ABSolute:MID[1] <NR3><br>MEASUREMENT:REFLevel:ABSolute:MID[1]? |
| <b>Related Commands</b> | <a href="#">MEASUREMENT:REFLevel:METHod</a>   |
| <b>Arguments</b>        | <NR3> is the mid reference level, in volts. The default is 0.0 V.                   |

**Examples** MEASUREMENT:REFLEVEL:ABSOLUTE:MID1 .71 sets the mid reference level to 0.71 V.

MEASUREMENT:REFLEVEL:ABSOLUTE:MID? might return  
:MEASUREMENT:REFLEVEL:ABSOLUTE:MID 0.7100E+00 indicating that the absolute mid1 reference level is set to 0.71 V.

## MEASUREMENT:REFLEVEL:ABSOLUTE:MID2

Sets or returns the mid reference level for the "to" waveform when taking a delay measurement, and is the 50% reference level when [MEASUREMENT:REFLEVEL:METHOD](#) is set to Absolute. This command affects the results of delay measurements.

---

**NOTE.** *this command affects the associated reference level parameter for all MEASUREMENTS:IMMED and the four periodic measurements.*

---

**Group** Measurement

**Syntax** MEASUREMENT:REFLEVEL:ABSOLUTE:MID2 <NR3>  
MEASUREMENT:REFLEVEL:ABSOLUTE:MID2?

**Related Commands** [MEASUREMENT:REFLEVEL:METHOD](#)

**Arguments** <NR3> is the mid reference level, in volts. The default is 0.0 V.

**Examples** MEASUREMENT:REFLEVEL:ABSOLUTE:MID2 0.5 sets the mid reference level for the delay waveform to 0.5 V.

MEASUREMENT:REFLEVEL:ABSOLUTE:MID2? might return  
:MEASUREMENT:REFLEVEL:ABSOLUTE:MID2 0.5000E+00 indicating that the absolute mid2 reference level is set to 0.5 V.

## MEASUREMENT:REFLEVEL:ABSOLUTE:MID<x>

Sets or returns the mid reference level for channel <x>, where x is the measurement channel.

**Group** Measurement

**Syntax**    `MEASUREMENT:REFLEVEL:ABSOLUTE:MID<x> <NR3>`  
`MEASUREMENT:REFLEVEL:ABSOLUTE:MID<x>?`

**Arguments**    <NR3> is the mid reference level in volts.

## MEASUREMENT:REFLEVEL:METHOd

Specifies or returns the reference level units used for measurement calculations.

---

**NOTE.** *this command affects the associated reference level parameter for all MEASUREMENTS:IMMED and the eight periodic measurements. To change the parameter for individual measurements, use the MEASUREMENT:MEAS<x>:REFLEVEL commands.*

---

**Group**    Measurement

**Syntax**    `MEASUREMENT:REFLEVEL:METHOd {ABSOLUTE|PERCENT}`  
`MEASUREMENT:REFLEVEL:METHOd?`

**Arguments**    **ABSOLUTE** specifies that the reference levels are set explicitly using the MEASUREMENT:REFLEVEL:ABSOLUTE commands. This method is useful when precise values are required (for example, when designing to published interface specifications, such as RS-232-C).

**PERCENT** specifies that the reference levels are calculated as a percent relative to HIGH and LOW. The percentages are defined using the MEASUREMENT:REFLEVEL:PERCENT commands.

**Examples**    `MEASUREMENT:REFLEVEL:METHOd ABSOLUTE` specifies that explicit user-defined values are used for the reference levels.

`MEASUREMENT:REFLEVEL:METHOd?` might return  
`:MEASUREMENT:REFLEVEL:METHOd PERCENT` indicating that the reference level units used are calculated as a percent relative to HIGH and LOW.

## MEASUREMENT:REFLEVEL:PERCENT:HIGH

Sets or returns the percent (where 100% is equal to HIGH) used to calculate the high reference level when `MEASUREMENT:REFLEVEL:METHOd` is set to Percent. This command affects the results of rise and fall measurements.



**NOTE.** *This command affects the associated reference level parameter for all MEASurements:IMMed and the four periodic measurements.*

**Group** Measurement

**Syntax** MEASUREMENT:REFLevel1:PERCent:HIGH <NR3>  
MEASUREMENT:REFLevel1:PERCent:HIGH?

**Related Commands** [MEASUREMENT:REFLevel:METHOD](#), [MEASUREMENT:IMMed:TYPE](#),  
[MEASUREMENT:MEAS<x>:TYPE](#)

**Arguments** <NR3> is the high reference level, ranging from 0 to 100%. The default high reference level is 90%.

**Examples** MEASUREMENT:REFLEVEL:PERCENT:HIGH 95 sets the high reference level to 95% of HIGH.

MEASUREMENT:REFLEVEL:PERCENT:HIGH? might return  
:MEASUREMENT:REFLEVEL:PERCENT:HIGH 90 indicating that the percentage high reference level is set to 90% of HIGH.

## MEASUREMENT:REFLevel:PERCent:LOW

Sets or returns the percent (where 100% is equal to HIGH) used to calculate the low reference level when [MEASUREMENT:REFLevel:METHOD](#) is set to Percent. This command affects the results of rise and fall measurements.

**NOTE.** *this command affects the associated reference level parameter for all MEASurements:IMMed and the four periodic measurements.*

**Group** Measurement

**Syntax** MEASUREMENT:REFLevel1:PERCent:LOW <NR3>  
MEASUREMENT:REFLevel1:PERCent:LOW?

**Related Commands** [MEASUREMENT:REFLevel:METHOD](#), [MEASUREMENT:IMMed:TYPE](#),  
[MEASUREMENT:MEAS<x>:TYPE](#)

|                  |  |
|------------------|--|
| <b>Arguments</b> | <NR3> is the low reference level, ranging from 0 to 100%. The default low reference level is 10%.  |
| <b>Examples</b>  | <p>MEASUREMENT:REFLEVEL:PERCENT:LOW 15 sets the high reference level to 15% of HIGH.</p> <p>MEASUREMENT:REFLEVEL:PERCENT:LOW? might return<br/>:MEASUREMENT:REFLEVEL:PERCENT:LOW 10 indicating that the percentage high reference level is set to 10% of HIGH.</p> |

## MEASUREMENT:REFLEVEL:PERCENT:MID[1]

Sets or returns the percent (where 100% is equal to HIGH) that is used to calculate the mid reference level when MEASUREMENT:REFLEVEL:METHOD is set to Percent. This command affects the results of period, frequency, delay, and all cyclic measurements.

---

**NOTE.** *this command affects the associated reference level parameter for all MEASUREMENTS:IMMED and the four periodic measurements.*

---

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Measurement  |
| <b>Syntax</b>           | <p>MEASUREMENT:REFLEVEL:PERCENT:MID[1] &lt;NR3&gt;<br/>MEASUREMENT:REFLEVEL:PERCENT:MID[1]?</p>  |
| <b>Related Commands</b> | <a href="#">MEASUREMENT:REFLEVEL:METHOD</a>  |
| <b>Arguments</b>        | <NR3> is the mid reference level, ranging from 0 to 100%. The default mid reference level is 50%.  |
| <b>Examples</b>         | <p>MEASUREMENT:REFLEVEL:PERCENT:MID 1 60 sets the mid reference level to 60% of HIGH.</p> <p>MEASUREMENT:REFLEVEL:PERCENT:MID? might return<br/>:MEASUREMENT:REFLEVEL:PERCENT:MID 65 indicating that the percentage mid reference level is set to 65% of HIGH.</p> |

## MEASUREMENT:REFLEVEL:PERCENT:MID2

Sets or returns the percent (where 100% is equal to HIGH) that is used to calculate the mid reference level for the second waveform specified when

**MEASUREMENT:REFLEVEL:METHOD** is set to Percent. This command affects the results of delay measurements.

---

**NOTE.** *this command affects the associated reference level parameter for all MEASUREMENTS:IMMED and the four periodic measurements.*

---

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Measurement   |
| <b>Syntax</b>           | MEASUREMENT:REFLEVEL:PERCENT:MID2 <NR3><br>MEASUREMENT:REFLEVEL:PERCENT:MID2?   |
| <b>Related Commands</b> | <a href="#">MEASUREMENT:REFLEVEL:METHOD</a>   |
| <b>Arguments</b>        | <NR3> is the mid reference level, ranging from 0 to 100%. The default mid reference level is 50%.   |
| <b>Examples</b>         | MEASUREMENT:REFLEVEL:PERCENT:MID2 40 sets the mid2 reference level to 40% of HIGH.<br><br>MEASUREMENT:REFLEVEL:PERCENT:MID2? might return<br>:MEASUREMENT:REFLEVEL:PERCENT:MID2 45 indicating that the percentage mid2 reference level is set to 45% of HIGH. |

## MEASUREMENT:REFLEVEL:PERCENT:MID<x>

Sets or returns the mid reference level for channel <x>, where x is the measurement channel.

|                  |   |
|------------------|---|
| <b>Group</b>     | Measurement   |
| <b>Syntax</b>    | MEASUREMENT:REFLEVEL:PERCENT:MID<x> <NR3><br>MEASUREMENT:REFLEVEL:PERCENT:MID<x>? |
| <b>Arguments</b> | <NR3> is the mid reference level in percent.                                      |

## MEASUREMENT:SNAPSHOT (No Query Form)

Displays the measurement snapshot list on the oscilloscope screen. The list contains the immediate values for all available measurements of the active signal.

**Group** Measurement

**Syntax** MEASUREMENT:SNAPSHOT

## MESSAGE

This command sets or queries message parameters.

**Group** Miscellaneous

**Syntax** MESSAGE  
MESSAGE?

**Examples** MESSAGE? might return MESSAGE:SHOW "TP401";BOX  
271,82,292,114;STATE 0 indicating that the message parameters.

## MESSAGE:BOX

Sets or returns the size and position of the message window. This command does not display the message unless MESSAGE:STATE is on.

X1 and Y1 are the screen coordinates of the top left corner of the message box. X2 and Y2 are the screen coordinates of the bottom right corner of the message box. All four coordinates are returned by the query.

Changing the text in the message box, using the MESSAGE:SHOW command, automatically resizes the message box. If you want a custom message box size, send the MESSAGE:BOX command after changing the text using the MESSAGE:SHOW command.

Message box settings and data are saved and restored in saved setups.

**Group** Display

**Syntax** MESSAGE:BOX <X1>, <Y1> [ , <X2>, <Y2> ]  
MESSAGE:BOX?

**Related Commands** [MESSAGE:STATE](#), [MESSAGE:SHOW](#), [MESSAGE:CLEAR](#)

**Arguments** <X1> and <X2> = 0 to 1023, and are pixel positions along the horizontal axis. <X1> defines the left and <X2> defines the right side of the window.

<Y1> and <Y2> = 0 to 767, and are pixel positions along the vertical axis. <Y1> defines the top and <Y2> defines the bottom of the window. The reserved height of all characters is 16 pixels so the window must be at least that high to fully display characters. <X2> and <Y2> are optional because the `MESSAGE:SHOW` command automatically sizes the box to fit the message. All four values are returned in a query.

## MESSAge:CLEAR (No Query Form)

Removes the message text from the message window.

**Group** Display

**Syntax** `MESSAge:CLEAR`

**Related Commands** [MESSAge:BOX](#), [MESSAge:SHOW](#), [MESSAge:STATE](#)

**Examples** `MESSAge:CLEAR`  
removes the message from the message window.

## MESSAge:SHOW

Clears the contents of the message window and displays the new message in the window. Changing the string length causes the message box to automatically resize to fit the text. The box may be resized using the [MESSAge:BOX](#) command.

**Group** Display

**Syntax** `MESSAge:SHOW <QString>`  
`MESSAge:SHOW?`

**Related Commands** [MESSAge:BOX](#), [MESSAge:CLEAR](#), [MESSAge:STATE](#)

**Arguments** <QString> is the message and can include any of the characters shown in the Character Set, Appendix A. The maximum length of the message is 1000 characters; the instrument ignores longer strings.

The message box size is set to fit the message. You can also set the message area height and width using the [MESSAge:BOX](#) command. The length of the message

that fits in the message area depends on the contents of the message because the width of characters varies.

If the message exceeds the limits of the message box, either horizontally or vertically, the portion of the message that exceeds the limits will not be displayed. The message string itself is not altered. The entire message can be returned as a query response regardless of what is displayed in the message box.

The message is left-justified, and is displayed on a single line starting with the top most line in the window. A new line character can be embedded in the string to position the message on multiple lines. You can also use white space and tab characters to position the message within a line. Text which does not fit within the message box is truncated. Defining a message box text string erases any previously displayed text within the message box.

You can send a tab by transmitting a tab character (`\t` or `\x09`) followed characters representing the most significant eight bits followed by significant eight bits of a 16-bit number. The number specifies the position relative to the left margin of the message area. For example, to tab send TAB (`\t` or `\x09`), NUL (decimal 0), and CR (decimal 13).

For example, using hexadecimal escape sequences, MESSAGE:SHOW `'\x09\x01\x17Hello'` when sent as a command would cause the 'Hello' to be displayed starting at pixel position 279 relative to the left margin set by the MESSAGE:BOX command. If you want to display characters starting at position 279, then  $279 = 0x0117$ ; split the hexadecimal number into two characters `0x01` and `0x17` and send `\x09\x01\x17`.

Special characters which control decoration are two character sequences where the first character is an escape (`0x1b`) and the second character is as described below.

|           |   |
|-----------|---|
| Bit 7     | 0   |
| Bit 6     | If set, inverse video is toggled from current state and the following text is displayed in the new inverse state until the state is toggled again. Remaining bits are ignored |
| Bit 5     | If set, the color index in the four LSB's (bits 0 through 3) is applied to the foreground or background color depending on the fg/bg bit (bit 4).                             |
| Bit 4     | If set, color change is applied to the background, otherwise applies to the foreground.   |
| Bit 0 – 3 | Specifies the color index (0 through 15) to change color as specified below:  |
|           | Index 0 Black (background)  |
|           | Index 1 Yellow (Ch 1)   |
|           | Index 2 Cyan (Ch 2)   |
|           | Index 3 Magenta (Ch 3)  |
|           | Index 4 Green (Ch 4)  |
|           | Index 5 Red (math)  |
|           | Index 6 White (reference)   |

|          |  |
|----------|--|
| Index 7  | Orange   |
| Index 8  | Gray (Graticule)   |
| Index 9  | White (text)   |
| Index 10 | Tek blue   |
| Index 11 | Bright blue  |
| Index 12 | Undefined  |
| Index 13 | Blue   |
| Index 14 | Undefined  |
| Index 15 | Dark blue  |
| Bit 4    | If set, the foreground color is set to the default foreground color. |
| Bit 3    | If set, the background color is set to the default background color. |
| Bit 2    | Undefined  |
| Bit 1    | Undefined  |
| Bit 0    | Undefined  |

The ESC (escape) character followed by the @ character turns inverse video on or off and can be embedded in the message string. Example: “abcESC@defESC@ghi” specifies the string “abcdefghi” where the “def” portion is displayed in inverse video.

Example: “abcESC#defESC)ESC@ghi” specifies the string “abcdefghi” where the “def” portion appears in the channel 3 color (magenta) and the “ghi” portion appears in the normal text color except it’s in inverse video.

An alternate way to enter characters is octal escape sequences. This consists of a backslash followed by numerals in the standard C language printf fashion.

Another way to enter characters is \xnn where the nn is the hexadecimal value of the character to display.

An advantage of these methods is that any controller program can be used. Another advantage is it’s easy to access characters with the high bit set, that is, those characters with a decimal value greater than 127.

An alternate way to enter certain characters is with a backslash followed by a single character (following “standard” Unix) as described in the table below.

|   |   |
|---|---|
| n | Newline (carriage return and line feed)   |
| \ | Backslash (\ is required to get a backslash character)  |
| t | Horizontal tab; the next 2 characters specify the pixel column to tab to as explained earlier |

If a backslash is followed by an undefined special character, the backslash is ignored and the character following it is accepted as is.

---

**NOTE.** *The use of any escape codes other than those described above may produce unpredictable results.*

---

**Examples**    MESSage:SHOW “Hello World”  
displays “Hello world” in the upper left corner of the box (you can define the box size with the MESSAGE BOX command).

MESSage:SHOW “**←**@Hello World**←**@ ... hello”  
displays “Hello world ... hello” in the upper left corner of the box and the word “world” is displayed in inverse video. In this example, **←** stands for the escape character. The escape character may appear differently for you depending on your controller program.

## MESSage:STATE

Controls the display of the message window on the screen.

**Group**    Display

**Syntax**    MESSage:STATE {OFF|ON|<NR1>}  
MESSage:STATE?

**Related Commands**    [MESSage:BOX](#)  
[MESSage:SHOW](#), [MESSage:CLEAR](#)

**Arguments**    OFF or <NR1> = 0 removes the message window from the screen.  
ON or <NR1> ≠ 0 displays the message window and its contents on the screen.

## NEWpass (No Query Form)

This command changes the password that enables access to password protected data. The PASSWord command must be successfully executed before using this command or an execution error will be generated.

**Group**    Miscellaneous

**Syntax**    NEWpass <QString>



|                         |   |
|-------------------------|---|
| <b>Related Commands</b> | <a href="#">*PUD</a><br><a href="#">PASSWord</a>  |
| <b>Arguments</b>        | <QString> is the new password, which can contain up to 16 characters.                       |
| <b>Examples</b>         | NEWPASS "mypassword" creates a new password (mypassword) for accessing your protected data. |

## \*OPC

Generates the operation complete message in the Standard Event Status Register (SESR) when all pending commands that generate an OPC message are complete. The \*OPC? query places the ASCII character "1" into the output queue when all such OPC commands are complete. The \*OPC? response is not available to read until all pending operations finish. (See page 3-1, *Status and Events*.)

The \*OPC command allows you to synchronize the operation of the oscilloscope with your application program. (See page 3-7, *Synchronization Methods*.)

**Table 2-40: Commands that Generate an OPC Message**

| Operation                   | Command  |
|-----------------------------|--|
| Single sequence acquisition | <a href="#">ACQUIRE:STATE</a> {ON NR 1}<br><a href="#">AUXin:PRObe:DEGAUss</a> EXECute<br><a href="#">CH&lt;x&gt;:PRObe:DEGAUss</a> EXECute<br><a href="#">DIAG:STATE</a> EXECute<br><a href="#">RECALL:SETUp</a> <file path><br><a href="#">RECALL:WAVEform</a> <file path>, REF<x><br><a href="#">SAVE:IMAGe</a> <file path><br><a href="#">SAVE:SETUp</a> <file path><br><a href="#">SAVE:WAVEform</a> <wfm>, {REF<x>}<br><a href="#">TEKSecure</a> |
| Hard copy operation         | <a href="#">HARDCopy</a> START   |
| Calibration step            | <a href="#">CALibrate:FACTory</a> {START PREVIOUS CONTINUE}  |

**Group** Status and Error

**Syntax** \*OPC  
\*OPC?

**Related Commands** [BUSY?](#), [\\*WAI](#)

- Examples** \*OPC generates the operation complete message in the SESR at the completion of all pending OPC operations.
- \*OPC? might return 1 to indicate that all pending OPC operations are finished.

## PICTBridge:PAPERSize

Sets the output print paper size.

**Group** PictBridge

**Syntax** PICTBridge:PAPERSize  
{DEFLT|L|L2|HAGAKIPCARD|MM54BY86|MM100BY150  
|IN4BY6|IN8BY10|LETTER|IN11BY17|A0|A1|A2|A3|A4|A5|A6|A7|A8|A9|  
B0|B1|B2|B3|  
B4|B5|B6|B7|B8|B9|ROLL89MM|ROLL127MM|ROLL100MM|ROLL210MM}

**Arguments** DEFLT for a default paper size.

L for a paper size L

L2 for a paper size 2L.

HAGAKIPCARD for a paper size Hagaki.

MM54BY86 for a card paper size.

MM100BY150 for paper size of 100\*150 MM.

IN4BY6 for a paper size of 4\*6.

IN8BY10 for a paper size of 8\*10.

LETTER for a letter paper size.

IN11BY17 for a paper size of 11\*17.

A0 for a A0 paper size.

A1 for a A1 paper size.

A2 for a A2 paper size.

A3 for a A3 paper size.

A4 for a A4 paper size.

A5 for a A5 paper size.

A6 for a A6 paper size.

A7 for a A7 paper size.

A8 for a A8 paper size.  
 A9 for a A9 paper size.  
 B0 for a B0 paper size.  
 B1 for a B1 paper size.  
 B2 for a B2 paper size.  
 B3 for a B3 paper size.  
 B4 for a B4 paper size.  
 B5 for a B5 paper size.  
 B6 for a B6 paper size.  
 B7 for a B7 paper size.  
 B8 for a B8 paper size.  
 B9 for a B9 paper size.  
 ROLL89MM for a 89 MM Roll paper size.  
 ROLL127MM for a 127 MM Roll paper size.  
 ROLL100MM for a 100 MM Roll paper size.  
 ROLL210MM for a 210 MM Roll paper size.

**Examples** PICTBRIDGE:PAPERSIZEL sets the paper size to L.  
 PICTBRIDGE:PAPERSIZE? might return PICTBRIDGE:PAPERSIZE DEFLT indicating that the paper size is set to Default.

## PICTBridge:DATEPrint

Enables or disables printing the date on the print output.

**Group** PictBridge

**Syntax** PICTBridge:DATEPrint {DEFLT|OFF|ON}

**Arguments** DEFLT is the default setting of the printer.  
 ON enables the date print on the print output.  
 OFF disables the date print on the print output.

- Examples** PICTBRIDGE:DATEPRINT? might return PICTBRIDGE:DATEPRINT DEFLT indicating that the default setting of the printer.
- PICTBRIDGE:DATEPRINT ON enables the date print on the print output.

## PICTBridge:DEfault (No Query Form)

Sets the arguments for all PictBridge commands to their default values. The default values are the same as printer default settings.

- Group** PictBridge
- Syntax** PICTBridge:DEfault
- Examples** PICTBRIDGE:DEFAULT sets all the PictBridge commands to their default values.

## PICTBridge:IDPrint

Enables or disables printing the oscilloscope model and serial number on the print output.

- Group** PictBridge
- Syntax** PICTBridge:IDPrint {DEFLT|OFF|ON}  
PICTBridge:IDPrint?

### Related Commands

- Arguments** OFF disables the oscilloscope model and serial number print on the print output.
- ON enables the oscilloscope model and serial number print on the print output.
- Default is provided for compatibility with TDS instruments that support pictbridge.

- Examples** PICTBRIDGE:IDPRINT? might return PICTBRIDGE:IDPRINT 0 indicating that the ID print is disabled on the print output.
- PICTBRIDGE:IDPRINT ON enables the printing of oscilloscope model and serial number on the print output.

## PICTBridge:IMAGESize

Sets or returns the image print size.

**Group** PictBridge

**Syntax** PICTBridge:IMAGESize {DEFLT|IN2P5BY3P25|L|IN4BY6|L2|IN8BY10|L4|E|CARD|HAGAKIPC|CM6BY8|CM7BY10|CM9BY13|CM10BY15|CM13BY18|CM15BY21|CM18BY24|A4|LETTER}

**Arguments**

DEFLT for a default image print size.

IN2P5BY3P25 for a 2\_5\*3\_25 image print size.

L for a 3\_5\*5 image print size.

IN4BY6 for a 4\*6 image print size.

L2 for a 5\*7 image print size.

IN8BY10 for a 8\*10 image print size.

L4 for a 254 MM\*178 MM image print size.

E for a 110 MM\*74 MM image print size.

CARD for a 89 MM\*55MM image print size.

HAGAKIPC for a 100 MM\*148 MM image print size.

CM6BY8 for a 6 CM\*8 CM image print size.

CM7BY10 for a 7 CM\*10 CM image print size.

CM9BY13 for a 9 CM\*13 CM image print size.

CM10BY15 for a 10 CM\*15 CM image print size.

CM13BY18 or a 13 CM\*18 CM image print size.

CM15BY21 for a 15 CM\*21 CM image print size.

CM18BY24 for a 18 CM\*24 CM image print size.

A4 for a A4 image print size.

LETTER for a Letter image print size.

**Examples**

PICTBRIDGE:IMAGESIZE? might return PICTBRIDGE:IMAGESIZE DEFLT indicating that the default image print size.

PICTBRIDGE:IMAGESIZE CARD sets the image print size to 89 MM\* 55 MM.

## PICTBridge:PAPERType

Sets or returns the paper type.

**Group** PictBridge

**Syntax** PICTBridge:PAPERType {DEFLT|PLAIN|PHOTO|FASTPHOTO}

**Arguments** DEFLT for a default print paper type.  
PLAIN for a plain print paper type.  
PHOTO for a photo print paper type.  
FASTPHOTO for a fastphoto print paper type.

**Examples** PICTBRIDGE:PAPERTYPE? might return PICTBRIDGE:PAPERTYPE DEFLT indicating that the default paper type.

PICTBRIDGE:PAPERTYPE PLAIN sets the paper type to Plain.

## PICTBridge:PRINTQual

Sets or returns the output print quality.

**Group** PictBridge

**Syntax** PICTBridge:PRINTQual {DEFLT|NRMAL|FINE|DRAFT}

**Arguments** DEFLT for the default quality print.  
NRMAL for a normal quality print.  
FINE for a fine quality print.  
DRAFT for a draft quality print.

**Examples** PICTBRIDGE:PRINTQUAL? might return PICTBRIDGE:PRINTQUAL DEFLT indicating that the default quality print.

PICTBRIDGE:PRINTQUAL FINE sets the print quality to Fine.

## PASSWord(No Query Form)

Enables the \*PUD and NEWpass set commands. Sending PASSWord without any arguments disables these same commands. Once the password is successfully entered, the \*PUD and NEWpass commands are enabled until the oscilloscope is powered off, or until the FACTory command or the PASSWord command with no arguments is issued.

To change the password, you must first enter the valid password with the PASSWord command and then change to your new password with the NEWpass command. Remember that the password is case sensitive.

**Group** Miscellaneous

**Syntax** PASSWord <QString>

**Related Commands** [NEWpass](#), [\\*PUD](#)

**Arguments** <QString> is the password and can include up to 10 characters. The factory default password is “XYZZY” and is always valid.

**Examples** PASSWORD “XYZZY” enables the \*PUD and NEWpass set commands.  
PASSWORD disables the \*PUD and NEWpass set commands. You can still use the query version of \*PUD.

## \*PSC

Sets or returns the power-on status flag that controls the automatic power-on handling of the DESER, SRER, and ESER registers. When \*PSC is true, the DESER register is set to 255 and the SRER and ESER registers are set to 0 at power-on. When \*PSC is false, the current values in the DESER, SRER, and ESER registers are preserved in nonvolatile memory when power is shut off and are restored at power-on.

**Group** Status and Error

**Syntax** \*PSC {OFF|ON|NR1}>  
\*PSC?

**Related Commands** [DESE](#), [\\*ESE](#), [FACTory](#), [\\*RST](#), [\\*SRE](#)

**Arguments** OFF sets the power-on status clear flag to false.  
 ON sets the power-on status clear flag to true.  
 <NR1> = 0 sets the power-on status clear flag to false. This disables the power-on clear allowing the oscilloscope to possibly assert SRQ after power-on; any other value sets the power-on status clear flag to true, enabling the power-on status clear preventing any SRQ assertion after power on.

**Examples** \*PSC 0 sets the power-on status clear flag to false.  
 \*PSC? might return 1 to indicate that the power-on status clear flag is set to true.

## \*PUD

Sets or returns a string of Protected User Data. This data is protected by the PASSWord command. You can modify it only by first entering the correct password. This password is not necessary to query the data.

**Group** Status and Error

**Syntax** \*PUD {<Block>|<Qstring>}  
 \*PUD?

**Related Commands** [PASSWord](#)

**Arguments** <Block> is a block containing up to 300 ASCII characters.  
 <Qstring> is a string containing up to 300 ASCII characters.

**Examples** \*PUD #229This oscilloscope belongs to me stores the string "This oscilloscope belongs to me" in the user protected data area.  
 \*PUD? might return #221PROPERTY OF COMPANY X

## \*RCL (No Query Form)

This command restores the state of the oscilloscope from a copy of the settings stored in memory (The settings are stored using the \*SAV command).

**Group** Save and Recall



**Syntax** \*RCL <NR1>

**Related Commands** [FACTory](#), [\\*LRN?](#), [RECALL:SETUp](#), [\\*RST](#), [\\*SAV](#), [SAVE:SETUp](#)

**Arguments** <NR1> is a value in the range from 1 to 10, which specifies a saved setup storage location.

**Examples** \*RCL 3 restores the oscilloscope from a copy of the settings stored in memory location 3.

## RECALL:SETUp (No Query Form)

Restores the state of the oscilloscope from a copy of the settings stored in memory. The settings are stored using the \*SAV command.

**Group** Save and Recall

**Syntax** RECALL:SETUp {FACTory|<NR1>|<file path>}

**Related Commands** [FACTory](#), [\\*RCL](#), [\\*RST](#), [\\*SAV](#), [SAVE:SETUp](#), [FILESystem:CWD](#)

**Arguments** FACTory restores the factory setup.  
 <NR1> is a value in the range from 1 to 10, which specifies a saved setup storage location.

<file path> specifies a location for an oscilloscope setup file. <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>:/<dir>/<filename>.<extension> and one or <dir>s are optional. If you do not specify them, the oscilloscope will read the file from the default directory (see [FILESystem:CWD](#)). <filename> stands for a filename; the use of wildcard characters in filenames is not supported. Filename extensions are not required, but highly recommended.

**Examples** RECALL:SETUP FACTORY recalls (and makes current) the oscilloscope setup to its factory defaults.  
 RECALL:SETUP 2 recalls the oscilloscope setup from setup storage location 2.  
 RECALL:SETUP "TEK00000.SET" recalls the setup from the file TEK00000.SET in the default directory for setups (E:/tekScope/setups).

## RECALL:WAVEform (No Query Form)

This command (no query form) recalls a stored waveform to a reference location.

**Group** Save and Recall

**Syntax** RECALL:WAVEform <file path>,REF<x>

**Related Commands** [SAVE:WAVEform](#), [FILESystem:CWD](#), [FILESystem](#)

**Arguments** REF<x> specifies a location in internal reference memory. Reference memory location values range from 1 through 4.

<file path> specifies a location for an oscilloscope setup file. <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>:/<dir>/<filename>.<extension> and one or <dir>s are optional. If you do not specify them, the oscilloscope will read the file from the default directory (see [FILESystem:CWD](#)). <filename> stands for a filename; the use of wildcard characters in filenames is not supported. Filename extensions are not required, but highly recommended.

**Examples** RECALL:WAVEFORM "tek00000.isf",REF1 recalls the waveform stored in the file named tek00000.isf from the current directory for waveforms to the reference location 1.

## REF<x>? (Query Only)

Returns reference waveform data for the channel specified by <x>, where x is the reference channel number.

**Group** Vertical

**Syntax** REF<x>?

## REF<x>:DATE? (Query Only)

Returns the date that reference waveform data for channel <x> was copied into the internal reference memory, where x is the reference channel number.

**Group** Vertical

**Syntax** REF<x>:DATE?

## REF<x>:HORizontal:DELay:TIME

Sets or returns the horizontal delay time for reference waveform <x>, where x is the reference channel number. The delay time is expressed in seconds and is limited to  $\pm 5$  times the reference horizontal scale.

**Group** Vertical

**Syntax** REF<x>:HORizontal:DELay:TIME <NR3>  
REF<x>:HORizontal:DELay:TIME?

**Arguments** <NR3> is the delay time in seconds.

**Examples** REF2:HORIZONTAL:DELAY:TIME 4.0E-6 sets the horizontal delay time for the REF2 waveform to 4  $\mu$ s.

## REF<x>:HORizontal:SCALE

Sets or returns the horizontal scale for reference waveform <x>, where x is the reference channel number.

**Group** Vertical

**Syntax** REF<x>:HORizontal:SCALE <NR3>  
REF<x>:HORizontal:SCALE?

**Arguments** <NR3> is the horizontal scale in seconds.

**Examples** REF1:HORIZONTAL:SCALE? might return REF1:HORIZONTAL:SCALE 4.0E-4.

## REF<x>:LABel

Sets or returns the reference waveform label for the channel specified by <x>, where x is the reference channel number.

**Group** Vertical

**Syntax** REF<x>:LABe1 <Qstring>  
REF<x>:LABe1?

**Arguments** <Qstring> is an alpha-numeric string of text, enclosed in quotes, that contains the label text for the reference channel <x> waveform. The text string is limited to 30 characters.

**Examples** REF2:LABEL? might return :REF2:LABEL "Clk wfm 2".

## REF<x>:POSition

Sets or returns the vertical position for channel <x>, where x is the reference channel number.

**Group** Vertical

**Syntax** REF<x>:POSition <NR3>  
REF<x>:POSition?

**Arguments** <NR3> is the vertical position in volts.

## REF<x>:SCAlE

Sets or returns the vertical scale for the channel specified by <x>, where x is the reference channel number.

**Group** Save and Recall

**Syntax** REF<x>:SCAlE <NR3>  
REF<x>:SCAlE?

**Arguments** NR3 is the vertical scale in volts

## REF<x>:TIME? (Query Only)

Returns the time that reference waveform data was copied into the internal reference memory for reference channel <x>, where x is the reference channel number.

**Group** Vertical

**Syntax** REF<x>:TIME?

**Examples** REF2:TIME? might return "16:54:05".

## REF<x>:VERTical:POSition

Sets or returns the vertical position of the reference waveform specified by <x>, where x is the reference channel number.

Increasing the position value of a waveform causes the waveform to move up, and decreasing the position value causes the waveform to move down. Position adjusts only the display position of a waveform. The position value determines the vertical graticule coordinate at which signal values are displayed. For example, if the position for Reference 3 is set to 2.0, the signal represented by that reference will be displayed at 2.0 divisions above the center of the screen.

**Group** Vertical

**Syntax** REF<x>:VERTical:POSition <NR3>  
REF<x>:VERTical:POSition?

**Related Commands** [CH<x>:POSition](#), [MATH\[1\]:VERTical:POSition](#)

**Arguments** <NR3> is the desired position, in divisions from the center horizontal graticule. The range is from -5.0 to 5.0 divisions.

**Examples** REF2:VERTICAL:POSITION 1.3E+00 positions the Reference 2 input signal 1.3 divisions above the center horizontal graticule.

REF1:VERTICAL:POSITION? might return :REF1:VERTICAL:POSITION -1.3000E+00 indicating that the current position of Reference 1 is 1.3 divisions below the center horizontal graticule.

## REF<x>:VERTical:SCAle

Sets or returns the vertical scale for the reference waveform specified by <x>, where x is the reference channel number.

Each waveform has a vertical scale parameter. For a signal with constant amplitude, increasing the Scale causes the waveform to be displayed smaller. Decreasing the scale causes the waveform to be displayed larger.

Scale affects all waveforms, but affects reference and math waveforms differently from channel waveforms:

- For reference and math waveforms, this setting controls the display only, graphically scaling these waveforms and having no affect on the acquisition hardware.
- For channel waveforms, this setting controls the vertical size of the acquisition window as well as the display scale. The range and resolution of scale values depends on the probe attached and any other external factors you have specified.

**Group** Vertical

**Syntax** REF<x>:VERTical:SCAle <NR3>  
REF<x>:VERTical:SCAle?

**Related Commands** CH<x>:SCAle, MATH[1]:VERTical:SCAle

**Arguments** <NR3> is the gain in user units-per-division.

**Examples** REF2:VERTICAL:SCALE 100E-03 sets the Reference 2 scale to 100 mV per division.

REF2:VERTICAL:SCALE? might return :REF2:VERTICAL:SCALE  
1.0000e+00 indicating that the current vertical scale setting for Reference 2 is 1 V per division.

## REM (No Query Form)

Embeds a comment within programs as a means of internally documenting the programs. The oscilloscope ignores these embedded comment lines.

**Group** Miscellaneous

|                  |  |
|------------------|--|
| <b>Syntax</b>    | REM <QString>  |
| <b>Arguments</b> | <QString> is a string that can contain a maximum of 80 characters.             |
| <b>Examples</b>  | REM "This is a comment" is a comment string that the oscilloscope will ignore. |

## \*RST (No Query Form)

Resets the oscilloscope to the factory default settings. The \*RST command does not alter the following:

- Calibration data that affect device specifications
- The Output Queue
- The Service Request Enable Register setting
- The Power-on status clear flag setting
- Alias definitions
- Stored settings
- The \*PUD? Response
- Any of the values associated with the DATA command.
- Oscilloscope password

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Status and Error   |
| <b>Syntax</b>           | *RST   |
| <b>Related Commands</b> | <a href="#">FACTory</a> , <a href="#">RECALL:SETUp</a> , <a href="#">SAVe:SETUp</a> , <a href="#">*PSC</a> , <a href="#">*RCL</a> , <a href="#">*SAV</a> |
| <b>Arguments</b>        | None   |
| <b>Examples</b>         | *RST resets the oscilloscope settings to factory defaults.   |

## \*SAV (No Query Form)

Stores the state of the oscilloscope to a specified memory location. You can use the \*RCL command to restore the oscilloscope to this saved state at a later time.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Save and Recall   |
| <b>Syntax</b>           | *SAV <NR1>  |
| <b>Related Commands</b> | *RCL, RECALL:SETUp, SAVe:SETUp  |
| <b>Arguments</b>        | <NR1> specifies a location in which to save the state of the oscilloscope. Location values range from 1 through 10. Using an out-of-range location value causes an execution error. Any settings that have been stored previously at this location will be overwritten. |
| <b>Examples</b>         | *SAV 2 saves the current oscilloscope state in memory location 2.   |

## SAVe:ASSIgn:TYPe

Sets or returns the assignment of the data to be saved when the front-panel Save button is pressed.

|                  |  |
|------------------|--|
| <b>Group</b>     | Save and Recall  |
| <b>Syntax</b>    | SAVe:ASSIgn:TYPe {IMAGE WAVEform SETUp}<br>SAVe:ASSIgn:TYPe?   |
| <b>Arguments</b> | IMAGE assigns the Save button to save screen images.<br>WAVEform assigns the Save button to save waveforms.<br>SETUp assigns the Save button to save setups. |

## SAVe:EVENTtable:BUS<x> (No Query Form)

Saves the data from bus<x> to a specified file and location; where x is the bus number

|               |                                    |
|---------------|------------------------------------|
| <b>Group</b>  | Save and Recall                    |
| <b>Syntax</b> | SAVe:EVENTtable:BUS<x> <file path> |



**Arguments** <file path> is a quoted string that defines the file name and path location where the event table will be stored.

---

**NOTE.** <filename> stands for a filename of up to 125 characters, followed by a period (".") and the three-character extension. Waveform files should have a .csv extension for comma-separated spreadsheet format files.

---

## SAVe:IMAGe (No Query Form)

Saves a capture of the screen image into the specified file. Supported image formats are PNG, Windows Bitmap, and TIFF. If an extension for a supported file type is added to the file name, then the corresponding format will be used. If no supported extension is added to the file, the format to use will be determined by the value obtained from the :SAVe:IMAGe:FILEFormat? query.

**Group** Save and Recall

**Syntax** SAvE:IMAGe <file path>

**Related Commands** [SAVe:ASSIgn:TYPe](#)

**Arguments** <file path> is a filename, including path, where the image will be saved. If you do not specify a directory, the oscilloscope will store the file in the current working directory. File name extensions are not required but are highly recommended. The images will be saved in E:/.

## SAVe:IMAGe:FILEFormat

Sets or returns the file format to use for saving screen images when the file type cannot be determined from the given file name or when screen images are captured by using the front panel.

**Group** Save and Recall

**Syntax** SAvE:IMAGe:FILEFormat {PNG|BMP|TIFF}  
SAVe:IMAGe:FILEFormat?

**Related Commands** [SAVe:IMAGe](#)

**Arguments** PNG saves the file in Portable Network Graphics format.  
 BMP saves the file in Microsoft Windows bitmap format.  
 TIFF saves the file in Tagged Image File Format.

## SAVe:IMAGe:LAYout

Sets or returns the layout to use for saved screen images.

**Group** Save and Recall

**Syntax** SAvE:IMAGe:LAYout {LANdScApE|PORTRaiT}  
 SAvE:IMAGe:LAYout?

**Arguments** LANdScApE specifies that screen images are saved in landscape format.  
 PORTRaiT specifies that screen images are saved in portrait format.

## SAVe:SETUp (No Query Form)

Stores the state of the oscilloscope to a specified memory location. You can later use the \*RCL command to restore the oscilloscope to this saved state.

**Group** Save and Recall

**Syntax** SAvE:SETUp {<file path>|<NR1>}

**Related Commands** [\\*RCL](#), [RECALL:SETUp](#), [\\*SAV](#)

**Arguments** <file path> is the target location for storing the setup file. <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>:<dir>/<filename>. <extension> and one or <dir>s are optional. If you do not specify them, the oscilloscope will store the file in the current working directory. <filename> stands for a filename. (Use of wildcard characters in filenames is not supported.) Filename extensions are not required but are highly recommended. For setups, use the extension ".SET".

<NR1> specifies a location for saving the current front-panel setup. The front-panel setup value ranges from 1 to 10. Using an out-of-range value causes an execution error. Any settings that have been stored previously at this location will be overwritten.

- Examples**     `SAVE:SETUP 5` saves the current oscilloscope setup in memory location 5.
- `SAVE:SETUP "TEK00000.SET"` saves the current oscilloscope setup in the file `TEK00000.SET` in the current working directory.

## SAVe:WAVEform (No Query Form)

This command saves a specified waveform or all displayed waveforms (excluding serial bus waveforms). Only individual analog waveforms (CH<x>, MATH and REF<x>) can be saved to reference memory locations.

You can save all displayed waveforms, excluding serial bus waveforms, to a single CSV file when the `SAVE:WAVEFORM:FILEFORMAT` is set to `SPREADSHEET`.

You can save all displayed waveforms, excluding serial bus waveforms to consecutive ISF (internal save format) files when the `SAVE:WAVEFORM:FILEFORMAT` is set to `INTERNAL`.

---

**NOTE.** *For MSO models, reduced resolution waveform data is not available for the digital channels.*

---

- Group**     Save and Recall
- Syntax**     `SAVE:WAVEform[<wfm>,{REF<x>}] | [<wfm>, <QString>] | [ALL, <QString>]`

**Related Commands**     [RECALL:WAVEform](#), [SAVe:WAVEform:FILEFormat](#)

- Arguments**
- `<wfm>`, `<REF<x>>` saves the specified waveform to the specified reference memory location. `<wfm>` can be any live analog channel (where `<x>` is the channel number), the MATH1 waveform, or another reference waveform (such as REF1).
- `<wfm>`, `<QString>` saves the specified waveform to the file specified in the quoted string argument. Any live channel (such as CH1), the MATH1 waveform, any reference waveform can be saved to a file.
- `ALL`, `<QString>` saves all displayed waveforms, excluding serial bus waveforms, to a single CSV file specified by the quoted string argument when the `SAVE:WAVEFORM:FILEFORMAT` is set to `SPREADSHEET`, or saves all displayed waveforms, excluding serial bus waveforms to individual ISF (internal save format) files with a file name prefix specified by the argument with an underscore (`_`) and the waveform ID (such as CH1, REF1, MATH) appended to the file name(s).

**Examples**     `SAVE:WAVEFORM CH1,REF1` saves the CH1 waveform in reference memory location 1.

`:SAVE:WAVEFORM:FILEFORMAT SPREADSHEET; :SAVE:WAVEFORM ALL, "E:/test_folder/test1_all.csv"` saves all displayed waveforms (excluding serial bus waveforms) to `E:/test_folder/test1_all.csv`.

`:SAVE:WAVEFORM:FILEFORMAT INTERNAL; :SAVE:WAVEFORM ALL, "E:/test_folder/test1"` saves all displayed waveforms (excluding serial bus waveforms) to individual files named `E:/test_folder/test1_<wfm>.isf` (for example `test1_CH1.isf`).

## SAVE:WAVEform:FILEFormat

Specifies or returns the file format for saved waveforms. Waveform header and timing information is included in the resulting file of non-internal formats. The oscilloscope saves DPO waveforms as a 500 x 200 matrix, with the first row corresponding to the most recently acquired data. The values specified by [DATA:START](#) and [DATA:STOP](#) determine the range of waveform data to output. In the event that [DATA:STOP](#) value is greater than the current record length, the current record length determines the last output value.

**Group**         Save and Recall

**Syntax**        `SAVE:WAVEform:FILEFormat {INTERNAL|SPREADSheet}`  
`SAVE:WAVEform:FILEFormat?`

**Related Commands**     [CURVe](#), [DATA](#), [DATA:START](#), [DATA:STOP](#), [SAVE:WAVEform](#), [WFMinpre:NR\\_Pt](#), [WFMOupre:NR\\_Pt?](#)

**Arguments**        `INTERNAL` specifies that waveforms are saved in an internal format, using a `.isf` filename extension. These files can be recalled as reference waveforms. When this argument is specified, the settings specified via the [DATA:START](#) and [DATA:STOP](#) commands have no meaning as the entire waveform is saved.

`SPREADSheet` specifies that waveform data is saved in a format that contains comma delimited values. These waveform data files are named using the `.csv` filename extension. Saving waveforms in CSV format enables spreadsheet programs to import the data.

**Examples**        `SAVE:WAVEFORM:FILEFORMAT INTERNAL` specifies that the internal file format is the format used for saving waveforms.

`SAVE:WAVEFORM:FILEFORMAT?` might return `:SAVE:WAVEFORM:FILEFORMAT INTERNAL` indicating that waveforms are saved using the internal format.

## SAVe:WAVEform:GATIng

Specifies whether save waveform operations should save the entire waveform (NONE) or a specified portion of the waveform.

**Group** Save and Recall

**Syntax** SAVE:WAVEform:GATIng {NONE|CURSors|SCREEN}  
SAVE:WAVEform:GATIng?

**Arguments** CURSors turns on cursors and the gates are the waveform record points at the cursor positions.

NONE saves the entire waveform.

SCREEN, if zoom is on, the gates are the start and end waveform record points of the zoom (upper) graticule, otherwise the gates are the start and end waveform record points of the main graticule.

**Examples** SAVE:WAVEFORM:GATING CURSors specifies that, when the waveform gating is set to cursors, save waveform operations should save the waveform points between the cursors. If cursors are turned off, waveform gating automatically reverts to NONE.

## SAVe:WAVEform:SPREADSheet:RESOLution

Specifies whether to save the full (LRL) or reduced (thumb) resolution waveform to a CSV file.

**Group** Waveform Enhanced

**Syntax** SAVE:WAVEform:SPREADSheet:RESOLution {FULL|REDUced}  
SAVE:WAVEform:SPREADSheet:RESOLution?

**Arguments** FULL specifies that full resolution waveform is saved to a CSV file.

REDUced specifies that reduced resolution waveform is saved to a CSV file.

**Examples** SAVE:WAVEFORM:SPREADSHEET:RESOLUTION? might return  
SAVE:WAVEFORM:SPREADSHEET:RESOLUTIONFULL indicating that full resolution waveform is saved to a CSV file.

SAVE:WAVEFORM:SPREADSHEET:RESOLUTIONREDUCed sets the reduced resolution waveform to be saved to a CSV file.

## SEARCH? (Query Only)

Returns all search-related settings.

|                 |  |
|-----------------|--|
| <b>Group</b>    | Search   |
| <b>Syntax</b>   | SEARCH?  |
| <b>Examples</b> | <p>SEARCH? might return:</p> <pre> :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:CONDITION SS; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:CONDITION SS; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:DATA:MOSI:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:DATA:MOSI:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:DATA:MISO:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:DATA:MISO:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:DATA:SIZE1; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:DATA:SIZE 1; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:CONDITION TXSTART; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:CONDITION TXSTART; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:RX:DATA:SIZE 1; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:RX:DATA:SIZE 1; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:RX:DATA:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:RX:DATA:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:TX:DATA:SIZE 1; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:TX:DATA:SIZE 1; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:TX:DATA:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:TX:DATA:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:CONDITION START; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:CONDITION START; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:DATA:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:DATA:VALUE "XXXXXXXX"; :SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:DATA:SIZE 1; :SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:DATA:SIZE 1; </pre> |

```

:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:ADDRESS:MODE ADDR7;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:ADDRESS:MODE ADDR7;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:ADDRESS:TYPE USER;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:ADDRESS:TYPE USER;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:ADDRESS:VALUE
"XXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:ADDRESS:VALUE
"XXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:CONDITION SOF;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:CONDITION SOF;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:FRAMETYPE DATA;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:FRAMETYPE DATA;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:VALUE "XXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:DATA:VALUE "XXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:QUALIFIER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:DATA:QUALIF IER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:IDENTIFIER:MODE
STANDARD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:IDENTIFIER:MODE
STANDARD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:IDENTIFIER:VALUE
"XXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:IDENTIFIER:VALUE
"XXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:CONDITION SYNCFIELD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:CONDITION SYNCFIELD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:VALUE"XXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:VALUE "XXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:HIVALUE
"XXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:HIVALUE
"XXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:QUALIFIER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:IDENTIFIER:VALUE
"XXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:IDENTIFIER:VALUE
"XXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:ERRTYPE SYNC;

```

```

:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:ERRTYPE SYNC;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:PARALLEL:VALUE
"XXXXXXXXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:PARALLEL:VALUE
"XXXXXXXXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:SOURCE
B1;:SEARCH:SEARCH1:TRIGGER:A:TYPE EDGE;
LEVEL 0.0E+0;LEVEL:CH1 0.0E+0; CH20.0E+0;CH30.0E+0;CH4
0.0E+0;MATH 0.0E+0;
REF1 0.0E+0 ;REF2
0.0E+0;:SEARCH:SEARCH1:TRIGGER:A:UPPERTHRESHOLD:CH1 1.4000;
CH2 1.4000; CH3 1.4000;CH4 1.4000;MATH 492.0000E-3;REF1
492.0000E-3;REF2 492.0000E-3;
:SEARCH:SEARCH1:TRIGGER:A:LOWERTHRESHOLD:CH1 0.0E+0;CH2
0.0E+0; CH30.0E+0;
CH4 0.0E+0;MATH 0.0E+0;REF1 0.0E+0;REF2 0.0E+0;
:SEARCH:SEARCH1:TRIGGER:A:EDGE:SOURCE CH1;SLOPE RISE;
:SEARCH:SEARCH1:TRIGGER:A:LOGIC:FUNCTION AND;
THRESHOLD:CH1 0.0E+0;CH2 0.0E+0; CH3 0.0E+0;CH4 0.0E+0;MATH
0.0E+0;REF1 0.0E+0; REF20.0E+0;
:SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:CH1 X;CH2 X;CH3 X;CH4
X;MATH X;REF1 X;REF2 X;
REF3 X;REF4 X;CLOCK:SOURCE NONE;EDGE RISE;
:SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:D0 X;D1 X;D2 X;
D3 X ;D4 X;D5 X;D6 X;D7 X;D8 X;D9 X;D10 X;D11 X;D12 X;D13
X;D14 X;D15 X;
:SEARCH:SEARCH1:TRIGGER:A:LOGIC:PATTERN:INPUT:CH1 X;CH2
X;CH3 X;
CH4 X;MATH X;REF1 X;REF2 X;REF3 X;REF4 X;D0 X;D1 X;D2 X;D3
X;D4 X;
D5 X;D6 X;D7 X ;D8 X;D9X;D10 X;D11 X;D12 X; D13 X;D14 X;D15
X;
:SEARCH:SEARCH1:TRIGGER:A:LOGIC:PATTERN:WHEN
TRUE;WHEN:LESSLIMIT 2.0000E-9;MORELIMIT 2.0000E-9;
:SEARCH:SEARCH1:TRIGGER:A:PULSEWIDTH:POLARITY POSITIVE;WHEN
LESSTHAN;WIDTH 8.0000E-9;
:SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY POSITIVE;WHEN
OCCURS;WIDTH 8.0000E-9;
:SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY POSITIVE;WHEN
SLOWER;DELTATIME 8.0000E-9;
:SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:SOURCE CH1;EDGE
RISE;THRESHOLD 0.0E+0;
:SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:SOURCE NONE;THRESHOLD
9.9100E+37;
:SEARCH:SEARCH1:TRIGGER:A:SETHOLD:HOLDTIME 2.000
0E-9;SETTIME 2.0000E-9;

```



```
THRESHOLD:CH1 0.0E+0;CH2 0.0E+0;CH3 0.0E+0;CH4 0.0E+0; MATH
0.0E+0;REF1 0.0E+0;REF2 0.0E+0;
:SEARCH:SEARCH1:STATE 0
```

## SEARCH:SEARCH<x>:COPY (No Query Form)

Copies the search criteria to the trigger, or the trigger criteria to a search. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:COPY {SEARCHtotrigger|TRIGgertosearch}

**Arguments** SEARCHtotrigger copies the search criteria to the trigger  
TRIGgertosearch copies the trigger criteria to the search

## SEARCH:SEARCH<x>:STATE

Sets the search state to on or off. <x> is the search number, which is always 1. The query form returns the search state.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:STATE {<NR1>|OFF|ON}  
SEARCH:SEARCH<x>:STATE?

**Arguments** OFF or <NR1> = 0 sets the search state to off.  
ON or <NR1> ≠ 0 sets the search state to on.

## SEARCH:SEARCH<x>:TOTAL? (Query Only)

Returns the total number of matches for the search. The total number of matches may be more than the number of marks placed. <x> is the search number, which is always 1.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TOTAL?

**Returns** <NR1> is the total number of matches.

## SEARCH:SEARCH<x>:TRIGger:A:BUS? (Query Only)

Queries the SEARCH:SEARCH<x>:TRIGger:A:BUS? settings.

**Group** Bus

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS?  
SEARCH:SEARCH<x>:TRIGger:A:BUS??

**Examples** SEARCH:SEARCH1:TRIGGER:A:BUS? might return

```

:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:CONDITION SS;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:CONDITION SS;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:DATA:MOSI:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:DATA:MOSI:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:DATA:MISO:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:DATA:MISO:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:SPI:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:SPI:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:CONDITION TXSTART;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:CONDITION TXSTART;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:RX:DATA:SIZE 1;
:SEARCH:SEARCH1 :TRIGGER:A:BUS:B2:RS232C:RX:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:RX:DATA:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:RX:DATA:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:TX:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:TX:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:RS232C:TX:DATA:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:RS232C:TX:DATA:VALUE "
XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:CONDITION START;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:CONDITION START;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:DATA:VALUE "XXXXXXXX";

```

```

:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:DATA:VALUE "XXXXXXXX";
:SEARCH1:TRIGGER:A:BUS:B1:I2C:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C: DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:ADDRESS:MODE ADDR7;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:ADDRESS:MODE ADDR 7;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:ADDRESS:TYPE USER;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:ADDRESS:TYPE USER;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:I2C:ADDRESS:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:I2C:ADDRESS:VALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:CONDITION SOF;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:CONDITION SOF;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:FRAMETYPE DATA;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:FRAMETYPE DATA;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:VALUE "XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:DATA:VALUE "XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2: CAN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:DATA:DIRECTION NOCARE;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:DATA:QUALIFIER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:DATA:QUALIFIER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:IDENTIFIER:MODE
STANDARD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:IDENTIFIER:MODE
STANDARD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:CAN:IDENTIFIER:VALUE
"XXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:CAN:IDENTIFIER:VALUE
"XXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:CONDITION SYNCFIELD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:CONDITION SYNCFIELD;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:SIZE 1;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:VALUE "XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:VALUE "XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:HIVALUE
"XXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:HIVALUE "XXXXXXXX
";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:DATA:QUALIFIER EQUAL;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:IDENTIFIER:VALUE
"XXXXXX";

```

```

:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:IDENTIFIER:VALUE
"XXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:ERRTYPE SYNC;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:LIN:ERRTYPE SYNC;
:SEARCH:SEARCH1:TRIGGER:A:BUS:B1:PARALLEL:VALUE
"XXXXXXXXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:B2:PARALLEL:VALUE
"XXXXXXXXXXXXXXXXXX";
:SEARCH:SEARCH1:TRIGGER:A:BUS:SOURCE B1

```

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:CONDition

Sets or returns the search condition for a CAN trigger search. SEARCH<x> is the search number and B<x> is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.   |
| <b>Group</b>      | Search   |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:CONDition<br>{SOF FRAMetype IDentifier DATA IDANDDATA EOF ACKMISS ERROR}<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:CONDition?   |
| <b>Arguments</b>  | <p>SOF specifies a search based on the start of frame.</p> <p>FRAMetype specifies a search based on the frame type.</p> <p>IDentifier specifies a search based on the frame identifier.</p> <p>DATA specifies a search based on the frame data.</p> <p>IDANDDATA specifies a search based on the frame identifier and data.</p> <p>EOF specifies a search base on the end of frame.</p> <p>ACKMISS specifies a search based on the missing ACK field.</p> <p>ERROR specifies a search based on a bit stuffing error.</p> |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:DATA:DIRection

Sets or returns the CAN search to be valid for Read, Write, or Either condition if the criteria is IDentifier. SEARCH<x> is the search number and B<x> is the bus number. This only applies if the search condition is IDentifier.

|                         |  |
|-------------------------|--|
| <b>Conditions</b>       | This command requires a DPO2AUTO application module.   |
| <b>Group</b>            | Search   |
| <b>Syntax</b>           | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:DATA:DIRection<br>{READ WRITE NOCARE}<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:DATA:DIRection? |
| <b>Related Commands</b> | <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:CONDItion</a>  |
| <b>Arguments</b>        | READ specifies the read condition.<br>WRITE specifies the write condition.<br>NOCARE specifies either a read or write condition.         |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier

Sets or returns the CAN data qualifier for a search. SEARCH<x> is the search number and B<x> is the bus number. This only applies if the trigger condition is IDANDDATA or DATA.

|                         |  |
|-------------------------|--|
| <b>Conditions</b>       | This command requires a DPO2AUTO application module.   |
| <b>Group</b>            | Search   |
| <b>Syntax</b>           | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier<br>{LESSThan MOREThan UNEQua1 LESSEQua1 MOREEQua1 EQua1}<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier?   |
| <b>Related Commands</b> | <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:VALue</a>   |
| <b>Arguments</b>        | LESSThan searches for bus data less than the value specified by <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:VALue</a> .<br>MOREThan searches for bus data more than the value specified by <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:VALue</a> .<br>UNEQua1 searches for bus data not equal to the value specified by <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:BUS:B&lt;x&gt;:CAN:DATA:VALue</a> . |

LESSEQUAL searches for bus data less than or equal to the value specified by [SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:VALUE](#).

MOREEQUAL searches for bus data more than or equal to the value specified by [SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:VALUE](#).

EQUAL searches for bus data equal to the value specified by [SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:VALUE](#).

## SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:SIZE

Sets or returns the length of the data string in bytes to be used for a CAN search if the search condition is DATA or IDANDDATA. SEARCH<x> is the search number and B<x> is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Search  |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:SIZE <NR1><br>SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:SIZE? |
| <b>Arguments</b>  | <NR1> is the data string length in bytes.   |

## SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:VALUE

Sets or returns the binary data string to be used for a CAN search if the search condition is ID or IDANDDATA. SEARCH<x> is the search number and B<x> is the bus number.

|                         |   |
|-------------------------|---|
| <b>Conditions</b>       | This command requires a DPO2AUTO application module.  |
| <b>Group</b>            | Search  |
| <b>Syntax</b>           | SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:VALUE <bin><br>SEARCH:SEARCH<x>:TRIGGER:A:BUS:B<x>:CAN:DATA:VALUE? |
| <b>Related Commands</b> | <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGGER:A:BUS:B&lt;x&gt;:CAN:DATA:QUALIFIER</a>                              |
| <b>Arguments</b>        | <bin> is the data in binary format.   |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:FRAMeType

Sets or returns the CAN Frame Type to be used if the trigger search condition is Frame Type. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:FRAMeType  
{DATA|REMOte|ERRor|OVERLoad}  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN:FRAMeType?

**Arguments** DATA specifies a data frame.  
REMOte specifies a remote frame.  
ERRor specifies an error frame.  
OVERLoad specifies an overload frame.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN{:IDentifier|:ADDRess}:MODE

Sets or returns the CAN addressing mode for a trigger search to a standard or extended format. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN{:IDentifier|:  
ADDRess}:MODE {STandard|EXTended}  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN{:IDentifier|:  
ADDRess}:MODE?

**Arguments** STandard specifies an 11-bit identifier field.  
EXTended specifies a 29-bit identifier field.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN{:IDentifier|:ADDRess}:VALue

Sets or returns the binary address string to be used for a CAN trigger search if the search condition is ID or IDANDDATA. SEARCH<x> is the search number and B<x> is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Search  |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN{:IDentifier :ADDRess}:VALue <bin><br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:CAN{:IDentifier :ADDRess}:VALue? |
| <b>Arguments</b>  | <bin> is the address in binary format.  |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:MODE

Sets or returns the I2C address mode to 7 or 10-bit. SEARCH<x> is the search number and B<x> is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.   |
| <b>Group</b>      | Search   |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:MODE {ADDR7 ADDR10}<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:MODE? |
| <b>Arguments</b>  | ADDR7 specifies 7-bit addresses.<br>ADDR10 specifies 10-bit addresses.   |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:TYPE

Sets or returns the I2C address type. SEARCH<x> is the search number and B<x> is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module. |
|-------------------|--|



**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:TYPe  
{GENeralcall|STARtbyte|HSmode|EEPROM|USER}  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:TYPe?

**Arguments** GENeralcall specifies the GENeralcall address type.  
STARtbyte specifies the STARtbyte address type.  
HSmode specifies the HSmode address type  
EEPROM specifies the EEPROM address type.  
USER specifies a user address.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:VALue

Sets or returns the binary address string to be used for an I2C trigger search if the search condition is ADDR or ADDRANDDATA. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:VALue <bin>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:ADDRess:VALue?

**Arguments** <bin> is the address in binary format.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:CONDition

Sets or returns the search condition for an I2C trigger search. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:CONDition  
 {START|STOP|REPEATstart|ACKMISS|ADDRESS|DATA|ADDRANDDATA}  
 SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:CONDition?

**Arguments** START specifies a search based on a start condition.  
 STOP specifies a search based on a stop condition.  
 REPEATstart specifies a search based on a repeat of start condition.  
 ACKMISS specifies a search based on a missing acknowledgement condition.  
 ADDRESS specifies a search based on an address.  
 DATA specifies a search based on a data condition.  
 ADDRANDDATA specifies a search based on an address and data condition.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:DIRection

Sets or returns the I2C search condition to be valid on a Read, Write, or Either condition. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:DIRection  
 {READ|WRITE|NOCARE}  
 SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:DIRection?

**Arguments** READ specifies a read condition.  
 WRITE specifies a write condition.  
 NOCARE specifies either a read or write condition.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:SIze

Sets or returns the length of the data string in bytes to be used for an I2C trigger search if the search condition is DATA or ADDRANDDATA. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:SIZE <NR1>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:SIZE?

**Arguments** <NR1> is the data string length in bytes.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:VALue

Sets or returns the binary data string to be used for an I2C trigger search if the search condition is DATA or ADDRANDDATA. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:VALue <bin>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:DATA:VALue?

**Arguments** <bin> is the data in binary format.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:CONDition

Sets or returns the search condition for a LIN search.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:CONDition  
{SYNCFieLd|IDentifier|DATA|IDANDDATA|WAKEup|SLEEP|ERROR}  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:CONDition?

**Arguments** SYNCFieLd specifies to search on the sync field.  
IDentifier specifies to search on the identifier.  
DATA specifies to search on the data.

IDANDDATA specifies to search on the identifier and the data.

WAKEup specifies to search on wake up.

SLEEP specifies to search on sleep.

ERROR specifies to search on errors.

**Examples** SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:CONDITION? might return  
SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:CONDITION SYNCFIELD  
indicating a search on the sync field.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue

Sets or returns the binary data string to be used for LIN searches if the search condition is ID or IDANDDATA.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue  
<QString>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue?

**Arguments** <QString> is a quoted string of 1s, 0s, or Xs representing the binary data string to be used for LIN searches if the search condition is ID or IDANDDATA.

**Examples** SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:HIVALUE? might return  
SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:HIVALUE "XXXXXXXX"  
indicating that the high value is don't care.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier

Sets or returns the LIN data qualifier. This only applies if the trigger condition is IDANDDATA or DATA.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

|                  |  |
|------------------|--|
| <b>Syntax</b>    | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier<br>{LESSThan MOREThan EQUa1 UNEQUa1 <br>LESSEQUa1 MOREEQUa1 INrange OUTrange}<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier?  |
| <b>Arguments</b> | LESSThan sets the LIN data qualifier to less than.<br>MOREThan sets the LIN data qualifier to greater than.<br>EQUa1 sets the LIN data qualifier to equal.<br>UNEQUa1 sets the LIN data qualifier to not equal.<br>LESSEQUa1 sets the LIN data qualifier to less than or equal.<br>MOREEQUa1 sets the LIN data qualifier to more than or equal.<br>INrange sets the LIN data qualifier to in range.<br>OUTrange sets the LIN data qualifier to out of range. |
| <b>Examples</b>  | SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER LESSThan<br>sets the data qualifier to lessthan.<br><br>SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER? might return<br>SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER EQUAL<br>indicating that the data qualifier is equal.  |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:SIZE

Sets or returns the length of the data string in bytes to be used for LIN Search, if search condition is DATA or IDANDDATA.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:SIZE <NR1><br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:SIZE? |
| <b>Arguments</b>  | <NR1> is the length of the data in bytes.   |
| <b>Examples</b>   | SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:SIZE 8 sets the LIN data<br>size is 8 bytes.                         |

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:SIZE? might return  
SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:SIZE 1 indicating that  
the LIN data size is 1 byte.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:VALue

Sets or returns the binary data string used for a LIN search if the search condition is ID or IDANDDATA.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.   |
| <b>Group</b>      | Bus  |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:VALue <QString><br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:DATA:VALue?  |
| <b>Arguments</b>  | <QString> is the binary data string for the search.  |
| <b>Examples</b>   | SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:VALUE? might return<br>SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:DATA:VALUE "XXXXXXXX"<br>indicating that the data value is don't care. |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:ERRTYPE

Sets or returns the error type used for a LIN Search.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:ERRTYPE<br>{SYNC PARity Checksum}<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:ERRTYPE? |
| <b>Arguments</b>  | SYNC specifies a sync error type.<br>PARity specifies a parity error type.<br>Checksum specifies a checksum error type.       |

**Examples** SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:ERRTYPE? might return  
SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:ERRTYPE SYNC indicating a  
SYNC error type.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue

Sets or returns the binary address string used for LIN search if search condition is ID or IDANDDATA.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue  
<QString>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue?

**Arguments** <QString> is a quoted string specifying the binary address string to be used for LIN search if search condition is ID or IDANDDATA.

**Examples** SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:IDENTIFIER:VALUE? might  
return SEARCH:SEARCH1:TRIGGER:A:BUS:B1:LIN:IDENTIFIER:VALUE  
"XXXXXX" indicating that the binary address is undefined.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:PARAllel:VALue

Sets or returns the binary data string to be used for a parallel trigger search.  
SEARCH<x> is the search number and B<x> is the bus number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:PARAllel:VALue <QString>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:PARAllel:VALue?

**Arguments** <QString> is the binary data string.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:CONDition

Sets or returns the condition for a RS232 trigger search. SEARCH<x> is the search number and B<x> is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2COMP application module.   |
| <b>Group</b>      | Search   |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:CONDition<br>{RXSTArt RXDATA RXENDPacket RXPARity TXSTArt TXDATA TXENDPacket TXPARity}<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:CONDition?   |
| <b>Arguments</b>  | <p>RXSTArt specifies a search based on the RX Start Bit.</p> <p>RXDATA specifies a search based on RX Data.</p> <p>RXENDPacket specifies a search based on the RX End of Packet condition.</p> <p>RXPARity specifies a search based on the RX parity.</p> <p>TXSTArt specifies a search base on the TX Start Bit.</p> <p>TXDATA specifies a search based on TX Data.</p> <p>TXENDPacket specifies a search based on the TX End of Packet condition.</p> <p>TXPARity specifies a search based on the TX parity.</p> |

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:SIZE

Sets or returns the length of the data string for a RS232 trigger search if the trigger condition is RX. SEARCH<x> is the search number and B<x> is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2COMP application module.  |
| <b>Group</b>      | Search  |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:SIZE<br>SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:SIZE? |
| <b>Arguments</b>  | <NR1> is the length of the data string in Bytes.  |



## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue

Sets or returns the binary data string for a RS232 trigger search if the condition involves RX. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue?

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE

Sets or returns the length of the data string to be used for a RS232 trigger search if the Trigger condition is TX. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE?

**Arguments** <NR1> is the length of the data string in Bytes.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue

Sets or returns the binary data string to be used for a RS232 trigger search if the condition involves TX. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue?

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:CONDition

Sets or returns the search condition for a SPI trigger search. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:CONDition  
{SS|MISO|MOSI|MISOMOSI}  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:CONDition?

**Arguments** SS specifies a search based on the Slave Selection condition.  
MISO specifies a search based on the Master-In Slave-Out condition.  
MOSI specifies a search based on the Master-Out Slave-In condition.  
MISOMOSI specifies a search based on the Master-In Slave-Out and Master-Out Slave-In conditions.

## SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{:MISO|:IN}:VALue

Sets or returns the binary data string for an SPI trigger search if the search condition is MISO or MISOMOSI. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{:MISO|:IN}:  
VALue <bin>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{:MISO|:IN}:  
VALue?

**Arguments** <bin> is the data string in binary format.

**SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{ :MOSI | :OUT } :VALue**

Sets or returns the binary data string for an SPI trigger search if search the condition is MOSI, or MISOMOSI. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{ :MOSI | :OUT } :VALue <bin>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA{ :MOSI | :OUT } :VALue?

**Arguments** <bin> is the data in binary format.

**SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA:SIZE**

Sets or returns the length of the data string for an SPI trigger search if the search condition is MISO, MOSI, or MISOMOSI. SEARCH<x> is the search number and B<x> is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA:SIZE <NR1>  
SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:DATA:SIZE?

**Arguments** <NR1> is the data string length in bytes.

**SEARCH:SEARCH<x>:TRIGger:A:BUS:SOUrce**

Sets or returns a bus serial search. <x> is the search number.

**Conditions** This command requires a DPO2AUTO or DPO2EMBD application module.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:BUS:SOURce {B1|B2}  
SEARCH:SEARCH<x>:TRIGger:A:BUS:SOURce?

**Arguments** B1 specifies the Bus 1 source.  
B2 specifies the Bus 2 source.

## SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe

Sets or returns the slope for an edge trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe {RISe|FALL}  
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe?

**Arguments** RISe specifies a rising edge.  
FALL specifies a falling edge.

## SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOURce

Sets or returns the source waveform for an edge trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOURce  
{CH1|CH2|CH3|CH4|MATH}  
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOURce?

**Arguments** CH<x> specifies one input channel as the edge source, where <x> is the channel number.  
MATH specifies the math waveform as the search source.

## SEARCH:SEARCH<x>:TRIGger:A:LEVel

Sets or returns the level for an edge trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LEVel {<NR3>|TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LEVel?

**Arguments** <NR3> specifies the trigger level, in volts.  
TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V

## SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x>

Sets or returns the level for an edge trigger search to determine where to place a mark. SEARCH<x> is the search number and CH<x> is the channel number. Each channel can have an independent level.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x> {<NR3>|TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x>?

**Arguments** <NR3> specifies the trigger level in volts.  
TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V

## SEARCH:SEARCH<x>:TRIGger:A:LEVel:MATH[1]

Sets or returns the math waveform level for an edge trigger search to determine where to place a mark. <x> is the search number. The value of MATH is 1 for all oscilloscopes.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LEVEl:MATH {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LEVEl:MATH?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V

## SEARCH:SEARCH<x>:TRIGger:A:LEVEl:REF<x>

Sets or returns the specified reference waveform level for an edge trigger search to determine where to place a mark. SEARCH<x> is the search number and REF<x> is the reference channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LEVEl:REF<x> {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LEVEl:REF<x>?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:FUNCTion

Sets or returns the logic operator for a logic trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:FUNCTion {AND|NAND}  
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:FUNCTion?

**Arguments** AND places a mark if all conditions are true.  
NAND places a mark if any of the conditions is false.

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:CH<x>

Sets or returns the Boolean logic criteria for a logic trigger search to determine where to place a mark. SEARCH<x> is the search number and CH<x> is the channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CH<x> {HIGH|LOW|X}  
SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CH<x>?

**Arguments** HIGH specifies the logic high.  
LOW specifies the logic low.  
X specifies a "don't care" state.

## SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:EDGE

Sets or returns whether the clock edge is a rising or falling for a logic search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:EDGE  
{FALL|RISe|EITHer}  
SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:EDGE?

**Arguments** RISe specifies a rising edge.  
FALL specifies a falling edge.  
EITHer specifies either a falling or rising edge.

## SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:SOURce

Sets or returns the clock source definition for a logic trigger search. <x> is the search number. If a clock source is defined, then the logic search is determined by the state of the other inputs at the clock transition. If no clock source is defined, then the logic search is determined only by the state of the inputs.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:SOURce  
{CH<x>|MATH|REF|D<x>|NONE}  
SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:CLOCK:SOURce?

- Arguments** CH<x> specifies a channel input as the clock source, where <x> = 1, 2, 3, or 4.  
 MATH specifies the math waveform as the clock source.  
 REF specifies the reference waveform as the clock source.  
 D<x> specifies the digital waveform as the clock source, where <x>=0–15  
 NONE specifies no clock source.

## SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:D<x>

Sets or returns the waveform logic criteria for a trigger search. SEARCH<x> is the search number and D<x> is the digital channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:D<x> {HIGH|LOW|X}  
 SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:D<x>?

- Arguments** HIGH specifies a high logic level.  
 LOW specifies a low logic level.  
 X specifies a “don’t care” condition.

## SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:MATH

Sets or returns the Boolean logic criteria for a logic trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:MATH {HIGH|LOW|X}  
 SEARCH:SEARCH<x>:TRIGger:A:LOGic:INPut:MATH?

- Arguments** HIGH specifies a high logic level.  
 LOW specifies a low logic level.  
 X specifies a “don’t care” condition.



**SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:REF<x>**

Sets or returns the Boolean logic criteria for a logic trigger search to determine where to place a mark. SEARCH<x> is the search number and REF<x> is the reference channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:REF<x> {HIGH|LOW|X}  
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:REF<x>?

**Arguments** HIGH specifies a high logic level.  
LOW specifies a low logic level.  
X specifies a “don’t care” condition.

**SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:CH<x>**

Sets or returns the logic criteria for a logic pattern trigger search to determine where to place a mark. SEARCH<x> is the search number and CH<x> is the channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:CH<x>  
{HIGH|LOW|X}  
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:CH<x>?

**Arguments** HIGH specifies a high logic level.  
LOW specifies a low logic level.  
X specifies a “don’t care” condition.

**SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:D<x>**

Sets or returns the logic criteria for a logic pattern search to determine where to place a mark. SEARCH<x> is the search number and D<x> is the digital channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:D<x>  
 {HIGH|LOW|X}  
 SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:D<x>?

**Arguments** HIGH specifies a high logic level.  
 LOW specifies a low logic level.  
 X specifies a “don’t care” condition.

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:MATH

Sets or returns the Boolean logic criteria for a logic pattern trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:MATH  
 {HIGH|LOW|X}  
 SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:MATH?

**Arguments** HIGH specifies a high logic level.  
 LOW specifies a low logic level.  
 X specifies a “don’t care” condition.

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:REF<x>

Sets or returns the Boolean logic criteria for a pattern trigger search to determine where to place a mark. SEARCH<x> is the search number and REF<x> is the reference channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:REF<x>  
 {HIGH|LOW|X}  
 SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:INPut:REF<x>?

**Arguments** HIGH specifies a high logic level.  
 LOW specifies a low logic level.  
 X specifies a “don’t care” condition.

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn

Sets or returns the condition for generating a logic pattern trigger search to determine where to place a mark. <x> is the search number.

|                  |  |
|------------------|--|
| <b>Group</b>     | Search   |
| <b>Syntax</b>    | SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn<br>{TRUE FALSE LESSThan MOREThan Than EQUa1 UNEQUa1}<br>SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn?   |
| <b>Arguments</b> | <p>TRUE places a mark when the pattern becomes true.</p> <p>FALSE places a mark when the pattern becomes false.</p> <p>LESSThan places a mark if the specific pattern is true less than the time set by the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit</a> command.</p> <p>MOREThan places a mark if the specific pattern is true more than the time set by the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit</a> command.</p> <p>EQUa1 places a mark if the specific pattern is true longer than the time set by the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit</a> command, but less than the specified time set by the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:MORELimit</a> command.</p> <p>UNEQUa1 places a mark if the specific pattern is true less than the time set by the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit</a> command, or longer than the specified time set by the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:MORELimit</a> command.</p> |

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit

Sets or returns the maximum time that the selected pattern may be true and still generate an A logic pattern search to place a mark. <x> is the search number.

|                  |  |
|------------------|--|
| <b>Group</b>     | Search   |
| <b>Syntax</b>    | SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit<br><NR3><br>SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit? |
| <b>Arguments</b> | <NR3> specifies the maximum amount of time to hold the pattern true.   |

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn:MORELimit

Sets or returns the minimum time that the selected pattern may be true and still generate an A logic pattern search to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn:MORELimit  
<NR3>  
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:WHEn:MORELimit?

**Arguments** <NR3> specifies the minimum amount of time to hold the pattern true.

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x>

Sets or returns the channel threshold level for a logic trigger search to determine where to place a mark. SEARCH<x> is the search number and CH<x> is the channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x>  
{<NR3>|TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x>?

**Arguments** <NR3> specifies the trigger level, in volts.  
TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH

Sets or returns the math waveform threshold level for a logic trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x>

Sets or returns the reference waveform threshold level for a logic trigger search to determine where to place a mark. SEARCH<x> is the search number and REF<x> is the reference channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x> {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x>?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:CH<x>

Sets or returns the channel waveform lower threshold to determine where to place a mark. This setting is applied to all channel searches that use a lower threshold. SEARCH<x> is the search number and CH<x> is the channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:CH<x> {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:CH<x>?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:MATH

Sets or returns the math waveform lower threshold to determine where to place a mark. This setting is applied to all math searches that use a lower threshold. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:MATH {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:MATH?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:REF<x>

Sets or returns the reference waveform lower threshold to determine where to place a mark. This setting is applied to all reference searches that use a lower threshold.  
SEARCH<x> is the search number and REF<x> is the reference channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:REF<x> {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:LOWerthreshold:REF<x>?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:POLarity

Sets or returns the polarity for a pulse trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:POLarity  
{NEGative|POSitive}  
SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:POLarity?

**Arguments** POSITIVE places a mark only when the polarity of the pulse is positive.  
NEGative places a mark only when the polarity of the pulse is negative.

## SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:SOURce

Sets or returns the source waveform for a pulse trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:SOURce  
{CH1|CH2|CH3|CH4|MATH|REF}  
SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:SOURce?

**Arguments** CH<x> specifies one input channel as the edge source, where <x> = 1, 2, 3 or 4.  
MATH specifies the math waveform as the search source.  
REF specifies the reference waveform as the search source.

## SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:WHEN

Sets or returns the condition for generating a pulse width search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:WHEN  
{LESSthan|MOREthan|EQua1|UNEQua1}  
SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:WHEN?

**Arguments** LESSthan places a mark if the pulse width is less than the time set by the [SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:WIDTH](#) command.  
MOREthan places a mark if the pulse width is more than the time set by the [SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:WIDTH](#) command.  
EQua1 places a mark if the pulse width is equal to the time set by the [SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:WIDTH](#) command within a tolerance of  $\pm 5\%$ .  
UNEQua1 places a mark if the pulse width is unequal to the time the time set by the [SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:WIDTH](#) command within a tolerance of  $\pm 5\%$ .

## SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:WIDth

Sets or returns the pulse width setting for a pulse width trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:WIDth <NR3>  
SEARCH:SEARCH<x>:TRIGger:A:PULSEwidth:WIDth?

**Arguments** <NR3> is the pulse width.

## SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity

Sets or returns the polarity setting for a runt trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity  
{EITHer|NEGative|POSitive}  
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity?

**Arguments** POSitive places a mark when the rising edge crosses the low threshold and the falling edge re-crosses the low threshold without either edge ever crossing the high threshold.

NEGative places a mark when the falling edge crosses the high threshold and the rising edge re-crosses the high threshold without either edge ever crossing the low threshold.

EITHer places a mark on a runt of either polarity.

## SEARCH:SEARCH<x>:TRIGger:A:RUNT:SOUrce

Sets or returns the source setting for a runt trigger search to determine where to place a mark. <x> is the search number.

**Group** Search



**Syntax** SEARCH:SEARCH<x>:TRIGger:A:RUNT:SOURce  
 {CH1|CH2|CH3|CH4|MATH|REF}  
 SEARCH:SEARCH<x>:TRIGger:A:RUNT:SOURce?

**Arguments** CH1–CH4 specifies an input channel as the edge source.  
 MATH specifies the math waveform as the search source.  
 REF specifies the reference waveform as the search source.

## SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn

Sets or returns the condition setting for a runt trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn  
 {LESSthan|MOREthan|EQua1|UNEQua1|OCCURS}  
 SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn?

**Arguments** OCCURS argument specifies a trigger event if a runt of any detectable width occurs.  
 LESSthan argument sets the oscilloscope to trigger if the a runt pulse is detected with width less than the time set by the [SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth](#) command.  
 MOREthan argument sets the oscilloscope to trigger if the a runt pulse is detected with width more than the time set by the [SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth](#) command.  
 EQua1 argument sets the oscilloscope to trigger when the pattern is true for a time period equal to the time period specified in [SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth](#) within a ±5% tolerance.  
 NOTEQua1 argument sets the oscilloscope to trigger when the pattern is true for a time period greater than or less than (but not equal) the time period specified in [SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth](#) within a ±5% tolerance.

## SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth

Sets or returns the width setting for a runt trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth <NR3>  
SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth?

**Arguments** <NR3> specifies the minimum width, in seconds.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:EDGE

Sets or returns the clock slope setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:EDGE {FALL|RISe}  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:EDGE?

**Arguments** FALL specifies polarity as the clock falling edge.  
RISe specifies polarity as the clock rising edge.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:SOURce

Sets or returns the clock source setting for an setup/hold trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:SOURce  
{CH1|CH2|CH3|CH4|MATH|REF}  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:SOURce?

**Related Commands** [SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:SOURce](#)

**Arguments** CH1–CH4 specifies an input channel as the edge source.  
MATH specifies the math waveform as the search source.  
REF specifies the reference waveform as the search source.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:THReshold

Sets or returns the clock threshold setting for an setup/hold trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:THReshold  
{<NR3>|TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:THReshold?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.  
<NR3> is the clock level, in volts.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:SOURce

Sets or returns the data source setting for an setup/hold trigger search to determine where to place a mark. <x> is the search number. You cannot specify the same source for both clock and data.

**Group** Search

**Syntax** DPO Models:  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:SOURce  
{CH1|CH2|CH3|CH4|MATH|REF|  
D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:SOURce?

**Related Commands** [SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCK:SOURce](#)

**Arguments** DPO Models:  
CH1-CH4 specifies an input channel as the search source.  
MATH specifies the math waveform as the search source.  
REF specifies the reference waveform as the search source.  
D0-D15 specifies the digital waveform as the search source.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:THReshold

Sets or returns the data threshold setting for an setup/hold trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:THReshold  
{<NR3>|TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:THReshold?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.  
<NR3> is the clock level, in volts.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:HOLDTime

Sets or returns the hold time setting for an setup/hold trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:HOLDTime <NR3>  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:HOLDTime?

**Arguments** <NR3> specifies the hold time setting in seconds. Positive values for hold time occur after the clock edge. Negative values occur before the clock edge.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:SETTime

Sets or returns the setup time setting for an setup/hold trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:SETTime <NR3>  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:SETTime?

**Arguments** <NR3> specifies the setup time for setup and hold violation triggering.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold:CH<x>

Sets or returns the trigger search setup and hold lower threshold to determine where to place a mark. Search<x> is the search number, which is always 1. CH<x> is the channel number.

**Conditions** MSO oscilloscopes only.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold:CH<x> <NR3>  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold:CH<x>?

**Arguments** <NR3> is the lower threshold in volts.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold{:MATH|:MATH1}

Sets or returns the trigger search setup and hold lower threshold to determine where to place a mark. Where <x> is the search number, which is always 1.

**Conditions** MSO oscilloscopes only.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold{:MATH|:MATH1}  
<NR3>  
SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold{:MATH|:MATH1}?

**Arguments** <NR3> is the lower threshold in volts.

## SEARCH:SEARCH<x>:TRIGger:A:SETHold:THReshold:REF<x>

Sets or returns the trigger search setup and hold lower threshold to determine where to place a mark. Search<x> is the search number, which is always 1. REF<x> is the reference waveform number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | MSO oscilloscopes only.   |
| <b>Group</b>      | Search  |
| <b>Syntax</b>     | SEARCH:SEARCH<x>:TRIGger:A:SETHo1d:THResho1d:REF<x> <NR3><br>SEARCH:SEARCH<x>:TRIGger:A:SETHo1d:THResho1d:REF<x>? |
| <b>Arguments</b>  | <NR3> is the lower threshold in volts.  |

### SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:DELTime

Sets or returns the transition time setting for an transition trigger search to determine where to place a mark. <x> is the search number.

|                  |  |
|------------------|--|
| <b>Group</b>     | Search   |
| <b>Syntax</b>    | SEARCH:SEARCH<x>:TRIGger:A{:TRANSition :RISEFall}:DELTime<br><NR3><br>SEARCH:SEARCH<x>:TRIGger:A{:TRANSition :RISEFall}:DELTime? |
| <b>Arguments</b> | <NR3> specifies the transition time, in seconds.   |

### SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:POLarity

Sets or returns the polarity setting for an transition trigger search to determine where to place a mark. <x> is the search number.

|                  |   |
|------------------|---|
| <b>Group</b>     | Search  |
| <b>Syntax</b>    | SEARCH:SEARCH<x>:TRIGger:A{:TRANSition :RISEFall}:POLarity<br>{EITHer NEGative POSitive}<br>SEARCH:SEARCH<x>:TRIGger:A{:TRANSition :RISEFall}:POLarity?   |
| <b>Arguments</b> | POSitive specifies that a pulse edge must traverse from the lower (most negative) to higher (most positive) level for transition triggering to occur.<br><br>NEGative specifies that a pulse edge must traverse from the upper (most positive) to lower (most negative) level for transition triggering to occur.<br><br>EITHer specifies either positive or negative polarity. |

## SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:SOUrce

Sets or returns the source setting for an transition trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:SOUrce  
{CH1|CH2|CH3|CH4|MATH}  
SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:SOUrce?

**Arguments** CH1–CH4 specifies one input channel as the edge source.  
MATH specifies the math waveform as the search source.

## SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:WHEN

Sets or returns the condition setting for an transition trigger search to determine where to place a mark. <x> is the search number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:WHEN  
{SLOWer|FASTer|EQua1|UNEQua1}  
SEARCH:SEARCH<x>:TRIGger:A{:TRANSition|:RISEFall}:WHEN?

**Arguments** FASTer sets the trigger to occur when the transitioning signal is faster than the set volts/second rate.  
SLOWer sets the trigger to occur when the transitioning signal is slower than the set volts/second rate.  
EQua1 sets the trigger to occur when the transitioning signal is equal to the set volts/second rate within a  $\pm 5\%$  tolerance.  
UNEQua1 sets the trigger to occur when the transitioning signal is not equal to the set volts/second rate  $\pm 5\%$ .

## SEARCH:SEARCH<x>:TRIGger:A:TYPE

Sets or returns the trigger type setting for a search to determine where to place a mark. <x> is the search number.

|                  |  |
|------------------|--|
| <b>Group</b>     | Search   |
| <b>Syntax</b>    | SEARCH:SEARCH<x>:TRIGger:A:TYPE<br>{EDGE SETHold PULSEwidth RUNT TRANSition LOGIC BUS (with the appropriate application module installed)}<br>SEARCH:SEARCH<x>:TRIGger:A:TYPE?   |
| <b>Arguments</b> | <p>RUNT triggers when a pulse crosses the first preset voltage threshold but does not cross the second preset threshold before recrossing the first. The thresholds are set with the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOWerthreshold:CH&lt;x&gt;</a> and <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:UPPerthreshold:CH&lt;x&gt;</a> commands.</p> <p>PULSEwidth triggers when a pulse is found that has the specified polarity and is either inside or outside the limits as specified by <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:LESSLimit</a> and <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:LOGIc:PATtern:WHEn:MORELimit</a>. The polarity is selected using the <a href="#">SEARCH:SEARCH&lt;x&gt;:TRIGger:A:RUNT:POLarity</a> command.</p> <p>TRANSition triggers when a pulse crosses both thresholds in the same direction as the specified polarity and the transition time between the two threshold crossings is greater or less than the specified time delta.</p> |

## SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:CH<x>

Sets or returns the channel waveform upper threshold to determine where to place a mark. This setting is applied to all channel searches that uses an upper threshold. SEARCH<x> is the search number and CH<x> is the channel number.

|                  |  |
|------------------|--|
| <b>Group</b>     | Search   |
| <b>Syntax</b>    | SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:CH<x> {TTL ECL}<br>SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:CH<x>?  |
| <b>Arguments</b> | <p>TTL specifies a preset TTL high level of 1.4 V.</p> <p>ECL specifies a preset ECL high level of -1.3 V.</p> |

## SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:MATH

Sets or returns the math waveform upper threshold to determine where to place a mark. This setting is applied to all math waveform searches that uses an upper threshold. <x> is the search number.



**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:MATH {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:MATH?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:REF<x>

Sets or returns the reference waveform upper threshold to determine where to place a mark. This setting is applied to all reference waveform searches that uses an upper threshold. SEARCH<x> is the search number and REF<x> is the reference channel number.

**Group** Search

**Syntax** SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:REF<x> {TTL|ECL}  
SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:REF<x>?

**Arguments** TTL specifies a preset TTL high level of 1.4 V.  
ECL specifies a preset ECL high level of -1.3 V.

## SElect

Sets or returns the selected waveform display (controlled by the front-panel) on or off.

**Group** Vertical

**Syntax** SElect {ON|OFF}

**Arguments** ON turns the selected waveform display on.  
OFF turns the selected waveform display off.

**Examples**     `SELECT?` might return the following

```
:SELECT:BUS1 0;BUS2 0;CH1 1;CH2 0;CH3 0;CH4 0;MATH 0;REF1
0;REF2 0;D0 0;D1 0;D2 0;D3 0;D4 0;D5 0;D6 0;D7 0;D8 0;D9
0;D10 0;D11 0;D12 0;D13 0;D14 0;D15 0,CONTROL CH1
```

## SElect:BUS<x>

This command turns on and off the display of the waveform for <x>, where x is the bus number. The query returns whether the channel is on or off but does not indicate whether it is the selected waveform.

**Group**     Vertical

**Syntax**     `SElect:BUS<x> {<NR1>|OFF|ON}`  
`SElect:BUS<x>?`

## SElect:CH<x>

Turns the display of the channel <x> waveform on or off, where <x > is the channel number. This command also resets the acquisition. The query returns whether the channel is on or off but does not indicate whether it is the selected waveform.

**Group**     Vertical

**Syntax**     `SElect:CH<x> {ON|OFF|<NR1>}`  
`SElect:CH<x>?`

**Arguments**     `ON` turns on the display of the specified waveform. This waveform also becomes the selected waveform.

`OFF` turns off the display of the specified waveform.

`<NR1> = 0` turns off the display of the specified waveform; any other value turns on the display of the specified waveform.

**Examples**     `SELECT:CH2 ON` turns the channel 2 waveform display on, and selects channel 2.

`SELECT:CH1?` might return `:SELECT:CH1 1` indicating that channel 1 is being displayed.

## SElect:CONTROI

Sets or returns the waveform that is the recipient of future channel-related commands, for example, the cursor commands. The command form also performs the equivalent of a **SElect:CH<x> ON** command, as well as the Math, Reference, and Bus variations of that command.

|                  |   |
|------------------|---|
| <b>Group</b>     | Vertical  |
| <b>Syntax</b>    | SElect:CONTROI {CH<x> MATH BUS<x>}<br>SElect:CONTROI?   |
| <b>Arguments</b> | <p>CH&lt;x&gt; specifies a channel waveform as the waveform affected by the front-panel controls. &lt;x&gt; is the channel number.</p> <p>MATH specifies the math waveform as the waveform that is affected by the front-panel controls.</p> <p>BUS&lt;x&gt; specifies a bus waveform as the waveform affected by the front-panel controls. &lt;x&gt; specifies the bus number.</p> |
| <b>Returns</b>   | NONE if all the channels are turned off. NONE is ignored on input.  |
| <b>Examples</b>  | <p>SELECT:CONTROL CH2 resets acquisition displays on channel 2, and causes the selected waveform to be the implied object of waveform commands.</p> <p>SELECT:CONTROL? might return :SELECT:CONTROL MATH indicating that math is the implied object of waveform commands.</p>   |

## SElect:D<x>

Turns on the display of the digital channel <x> and resets the acquisition. <x> is the channel number. The query returns whether the channel is on or off but does not indicate whether it is the selected waveform.

|                  |  |
|------------------|--|
| <b>Group</b>     | Vertical   |
| <b>Syntax</b>    | SElect:D<x> {<NR1> OFF ON}<br>SElect:D<x>?   |
| <b>Arguments</b> | ON turns on the display of the specified waveform. This waveform also becomes the selected waveform. |

OFF turns off the display of the specified waveform.

<NR1> = 0 turns off the display of the specified waveform; any other value turns on the display of the specified waveform.

**Examples**     `SELECT:D2 ON` turns the digital channel 2 waveform display on, and selects digital channel 2.

`SELECT:D2?` might return `:SELECT:D2 1` indicating that digital channel 2 is being displayed.

## SElect:MATH[1]

Turns on and off the display of the math waveform. The query returns whether the math waveform is on or off but does not indicate whether it is the selected waveform.

**Group**     Vertical

**Syntax**     `SElect:MATH[1] {ON|OFF|<NR1>}`  
`SElect:MATH[1]?`

**Arguments**     `ON` turns on the display of the specified waveform. This waveform also becomes the selected waveform.

`OFF` turns off the display of the specified waveform.

<NR1> = 0 turns off the display of the specified waveform; any other value turns on the display of the specified waveform.

**Examples**     `SELECT:MATH ON` turns the math waveform display on, and selects it.

`SELECT:MATH?` might return `:SELECT:MATH 1` indicating that the math waveform is being displayed.

## SElect:REF<x>

Turns on and off the display of the reference waveform <x>. The <x > variable represents the reference channel number. The query returns whether the channel is on or off.

**Group**     Vertical

|                  |   |
|------------------|---|
| <b>Syntax</b>    | <code>SElect:REF&lt;x&gt; {ON OFF &lt;NR1&gt;}</code><br><code>SElect:REF&lt;x&gt;?</code>  |
| <b>Arguments</b> | <p>ON turns on the display of the specified waveform. This waveform also becomes the selected waveform.</p> <p>OFF turns off the display of the specified waveform.</p> <p>&lt;NR1&gt; = 0 turns off the display of the specified waveform; any other value turns on the display of the specified waveform.</p> |
| <b>Examples</b>  | <p><code>SELECT:REF2 ON</code> turns the channel 2 waveform display on, and selects reference waveform 2.</p> <p><code>SELECT:REF2?</code> might return <code>:SELECT:REF2 1</code> indicating that reference waveform 2 is being displayed.</p>  |

## SET? (Query Only)

Returns the commands that list the oscilloscope settings except for configuration information for the calibration values, the [WFMinpre?](#) query, and the [WFMOutpre?](#) query. This query allows you to record or "learn" the current oscilloscope settings. You can use these commands to return the oscilloscope to the state it was in when you made the SET? query. The SET? query always returns command headers, regardless of the setting of the [HEADer](#) command. This is because the returned commands are intended to be sent back to the oscilloscope as a command string. The [VERBose](#) command can still be used to specify whether the returned headers should be abbreviated or full-length.

This command is identical to the [\\*LRN?](#) command.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Miscellaneous  |
| <b>Syntax</b>           | <code>SET?</code>  |
| <b>Related Commands</b> | <a href="#">HEADer</a> , <a href="#">*LRN?</a> , <a href="#">VERBose</a>   |
| <b>Examples</b>         | <p>SET? returns a long response, part of which could be as follows: <code>:SET :ACQUIRE:STOPAFTER RUNSTOP;STATE 1;MODE SAMPLE;NUMENV INFINITE;NUMAVG 16;SAMPLINGMODE RT;:HEADER 1;:LOCK NONE;:LANGUAGE ENGLISH;:VERBOSE 1;:ALIAS:STATE 0;: DISPLAY:COLOR:PALETTE NORMAL;:DISPLAY:PERSISTENCE 0.0000;CLOCK 1;GRATICULE FULL;INTENSITY:WAVEFORM 30;GRATICULE 75;BACKLIGHT HIGH;:HARDCOPY:INKSAVER</code></p> |

```

OFF;LAYOUT LANDSCAPE;PREVIEW 0; :SAVE:IMAGE:FILEFORMAT
BMP;:SAVE:WAVEFORM:FILEFORMAT INTERNAL;:SAVE:ASSIGN:TYPE
SETUP;:TRIGGER:A:MODE AUTO;TYPE EDGE;LEVEL
20.0000E-3;LEVEL:CH1 20.0000E-3;CH2 0.0000;CH3
0.0000;CH4 0.0000;:TRIGGER:A:UPPERTHRESHOLD:CH1
1.4000;CH2 800.0000E-3;CH3 800.0000E-3;CH4 800.0000E-3;:
TRIGGER:A:LOWERTHRESHOLD:CH1 20.0000E-3;CH2
0.0000;CH3 0.0000;CH4 0.0000;:TRIGGER:A:HOLDOFF:TIME
20.0000E-9;:TRIGGER:A:EDGE:SOURCE CH1;COUPLING
DC;SLOPE RISE;:TRIGGER:A:LOGIC:CLASS SETHOLD;FUNCTION
AND;THRESHOLD:CH1 20.0000E-3;CH2 0.0000;CH3
0.0000;CH4 0.0000;:TRIGGER:A:LOGIC:INPUT:CH1
X;CH2 X;CH3 X;CH4 X;CLOCK:SOURCE NONE;EDGE
RISE;:TRIGGER:A:LOGIC:PATTERN:INPUT:CH1 X;CH2
X;CH3 X;CH4 X;:TRIGGER:A :LOGIC:PATTERN:WHEN
TRUE;WHEN:LESSLIMIT 4.0000E-9;LIMIT 4.0000E-9;:TRIGGER:A
:SETHOLD:CLOCK:SOURCE CH1;EDGE RISE;THRESHOLD
20.0000E-3;:TRIGGER:A:SETHOLD:DATA:SOURCE CH2;
    
```

## SETUP<x>:DATE? (Query Only)

Returns the date when the oscilloscope setup was saved for the specified channel <x>.

**Group** Save and Recall

**Syntax** SETUP<x>:DATE?

**Examples** SETUP4:DATE? might return SETUP4:DATE: 04-18-06 which is the setup date for channel 4.

## SETUP<x>:LABEL

Sets or returns the setup label for the specified channel <x>.

**Group** Save and Recall

**Syntax** SETUP<x>:LABEL <Qstring>

**Arguments** <Qstring> is an alpha-numeric string of characters, enclosed in quotes, that defines the label text for SETUP<x>. The length of the string is limited to 30 characters.

**Examples** SETUP:LABEL? might return SETUP1:LABEL: TEST 2 which is the label setup for channel 1.

## SETUP<x>:TIME? (Query Only)

Returns the time when the oscilloscope setup was saved for the specified channel <x>.

**Group** Save and Recall

**Syntax** SETUP<x>:TIME?

**Examples** SETUP2:TIME? might return "SETUP2:TIME: 15:24:07 which is the setup time for channel 2.

## \*SRE

The \*SRE (Service Request Enable) command sets or returns the bits in the Service Request Enable Register. For information, refer to Registers.

**Group** Status and Error

**Syntax** \*SRE <NR1>  
\*SRE?

**Related Commands** [\\*CLS](#), [DESE](#), [\\*ESE](#), [\\*ESR?](#), [EVENT?](#), [EVMsg?](#), [FACTory](#), [\\*STB?](#)

**Arguments** <NR1> is a value in the range from 0 through 255. The binary bits of the SRER are set according to this value. Using an out-of-range value causes an execution error. The power-on default for SRER is 0 if \*PSC is 1. If \*PSC is 0, the SRER maintains the previous power cycle value through the current power cycle.

**Examples** \*SRE 48 sets the bits in the SRER to binary 00110000.  
 \*SRE? might return 32, showing that the bits in the SRER have the binary value of 00100000.

## \*STB? (Query Only)

\*STB? (Read Status Byte) returns the contents of the Status Byte Register (SBR) using the Master Summary Status (MSS) bit. For information, refer to Registers.

**Group** Status and Error

**Syntax** \*STB?

**Related Commands** \*CLS, DESE, \*ESE, \*ESR?, EVENT?, EVMsg?, FACTory, \*SRE

**Returns** <NR1>

**Examples** \*STB? might return 96, showing that the SBR contains the binary value 01100000.

## TEKSecure (No Query Form)

This command initializes both waveform and setup memories, overwriting any previously stored data. These are the [WFMinpre?](#), [WFMOupre?](#), [DATA](#), [DATA:RESOLution](#) and [DATA:COMPosition](#) command values after the TEKSecure operation.

```
:WFMOUPRE:BYT_NR 1 :WFMOUPRE:BIT_NR 8 :WFMOUPRE:ENCDG BIN
:WFMOUPRE:BN_FMT RI :WFMOUPRE:BYT_OR MSB :WFMOUPRE:WFID
"Ch1, DC coupling, 100.0mV/div, 4.000us/div, 10000 points,
Sample mode" :WFMOUPRE:NR_PT 10000 :WFMOUPRE:PT_FMT
Y :WFMOUPRE:XUNIT "s" :WFMOUPRE:XINCR 4.0000E-9
:WFMOUPRE:XZERO -20.0000E-6 :WFMOUPRE:PT_OFF 0
:WFMOUPRE:YUNIT "V" :WFMOUPRE:YMULT 4.0000E-3
:WFMOUPRE:YOFF 0.0000 :WFMOUPRE:YZERO 0.0000
:WFMINPRE:BYT_NR 1 :WFMINPRE:BIT_NR 8 :WFMINPRE:ENCDG BIN
:WFMINPRE:BN_FMT RI :WFMINPRE:BYT_OR MSB :WFMINPRE:NR_PT
10000 :WFMINPRE:PT_FMT Y :WFMINPRE:XUNIT "s" :WFMINPRE:XINCR
4.0000E-9 :WFMINPRE:XZERO 0.0000 :WFMINPRE:PT_OFF 0
:WFMINPRE:YUNIT "V" :WFMINPRE:YMULT 4.0000E-3 :WFMINPRE:YOFF
0.0000 :WFMINPRE:YZERO 0.0000 DATA:DESTINATION REF1
DATA:ENCDG RIBINARY DATA:SOURCE CH1 DATA:START 1 DATA:STOP
```



```
10000 DATA:WIDTH 1 :DATA:RESOLUTION FULL :DATA:COMPOSITION
COMPOSITE_YT
```

---

**NOTE.** *The TEKSecure command can take up to five minutes to complete. The oscilloscope is inoperable during this period.*

---

**Group** Miscellaneous

**Syntax** TEKSecure

**Examples** TEKSECURE initializes both waveform and setup memories.

This is a program example of how to generate an SRQ when TEKSECURE completes:

```
# Bit 0 of the DESE (Device Event Status Enable Register)
# enables OPC to be reported to the SESR (Standard Event
# Status Register)
DESE 255
# Bit 0 of the ESER (Event Status Enable Register)
# enables OPC to be summarized in the ESB (Event Status #
# Bit) of the SBR (Status Byte Register)
*ESE 255
# Bit 5 of the SRE (Service Request Enable Register)
enables
# the generation of SRQ when the ESB bit of the SBR
becomes # TRUE
*SRE 32
TEKSECURE;*OPC
```

When the TEKSECURE operation has completed, the OPC bit of the SESR will be TRUE and SRQ will have been generated.

## TIME

Sets or returns the time that the oscilloscope displays.

**Group** Miscellaneous

**Syntax** TIME <QString>  
TIME?

**Related Commands**    [DATE](#)

**Arguments**    <QString> is a time in the form "hh:mm:ss" where hh refers to a two-digit hour number, mm refers to a two-digit minute number from 00 to 59, and ss refers to a two-digit second number from 00 to 59.

**Examples**    TIME "14:00:00" sets the time to exactly 2:00 p.m.  
TIME? might return :TIME "14:05:17" indicating that the current time is set to 2:05 p.m. and 17 seconds.

## TOTALuptime? (Query Only)

This command returns the total number of hours that the oscilloscope has been powered on since the nonvolatile memory was last programmed (usually since the initial manufacturing process).

**Group**    Miscellaneous

**Syntax**    TOTALuptime?

## \*TRG (No Query Form)

Performs a group execute trigger on commands defined by \*DDT.

**Group**    Miscellaneous

**Syntax**    \*TRG

**Related Commands**    [\\*DDT](#)

**Examples**    \*TRG immediately executes all commands that have been defined by \*DDT.

## TRIGger (No Query Form)

Forces a trigger event to occur.

**Group**    Trigger

|                  |  |
|------------------|--|
| <b>Syntax</b>    | TRIGger FORCE<br>TRIGger?  |
| <b>Arguments</b> | FORCE creates a trigger event. If TRIGger:STATE is set to READy, the acquisition will complete. Otherwise, this command will be ignored. |
| <b>Examples</b>  | TRIGGER FORCE forces a trigger event to occur.   |

## TRIGger:FREQuency? (Query Only)

Returns the available trigger frequency in hertz. If the trigger frequency is not currently available, the IEEE Not A Number (NaN = 9.9100E+37) value is returned. The maximum precision of the returned frequency is 6 digits.

|               |                     |
|---------------|---------------------|
| <b>Group</b>  | Trigger             |
| <b>Syntax</b> | TRIGger :FREQuency? |

## TRIGger:A

Sets the A trigger level automatically to 50% of the range of the minimum and maximum values of the trigger input signal. The query returns current A trigger parameters. The trigger level is the voltage threshold through which the trigger source signal must pass to generate a trigger event. This command works for the following cases: Edge Trigger (when source is Not Line), Logic Trigger (when Clock Source is not Off or Logic Pattern is Don't Care), and Pulse Width Trigger.

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Trigger  |
| <b>Syntax</b>           | TRIGger:A SETLeve1<br>TRIGger:A?   |
| <b>Related Commands</b> | <a href="#">TRIGger:A:EDGE?</a> , <a href="#">TRIGger:A:LOGIc?</a> , <a href="#">TRIGger:A:PULSe?</a>                          |
| <b>Arguments</b>        | SETLeve1 sets the A trigger level to 50% of the range of the minimum and maximum values of the trigger input signal.           |
| <b>Examples</b>         | TRIGGER:A SETLEVEL sets the A trigger level to 50% of the range of the minimum and maximum values of the trigger input signal. |

```

TRIGGER:A? might return a long response with A trigger parameters,
some of which could be as follows: :TRIGGER:A:MODE AUTO;TYPE
EDGE;LEVEL 20.0000E-3;LEVEL:CH1 20.0000E-3;CH2 0.0000;
CH3 0.0000;CH4 0.0000;:TRIGGER:A:UPPERTHRESHOLD:CH1
1.4000;CH2 800.0000E-3;CH3 8 00.0000E-3;CH4
800.0000E-3;:TRIGGER:A:LOWERTHRESHOLD:CH1 20.0000E-3;CH2
0.0000;CH3 0.0000;CH4 0.0000;:TRIGGER:A:HOLDOFF:TIME
20.0000E-9;:TRIGGER:A:EDGE:SOURCE CH1;COUPLING
DC;SLOPE RISE;:TRIGGER:A:LOGIC:CLASS SETHOLD;FUNCTION
AND;THRESHOLD: CH1 20.0000E-3;CH2 0.0000;CH3
0.0000;CH4 0.0000;:TRIGGER:A:LOGIC:INPUT:CH1
X;CH2 X;CH3 X;CH4 X;CLOCK:SOURCE NONE;EDGE
RISE;:TRIGGER:A:LOGIC:PATTERN:INPUT:CH1 X; CH2
X;CH3 X;CH4 X;:TRIGGER:A:LOGIC:PATTERN:WHEN
TRUE;WHEN:LESSLIMIT 4.0000E-9;MO RELIMIT
4.0000E-9;:TRIGGER:A:SETHOLD:CLOCK:SOURCE CH1;EDGE
RISE;THRESHOLD 20.000 0E-3;:TRIGGER:A:SETHOLD:DATA:SOURCE
CH2;THRESHOLD 0.0000;:TRIGGER:A:SETHOLD:HOLD TIME
4.0000E-9;SETTIME 4.0000E-9;:TRIGGER:A:PULSE:CLASS
TRANSITION;:TRIGGER:A:PULSEWIDTH:SOURCE
CH1;POLARITY POSITIVE;WHEN LESSTHAN;WIDTH
4.0000E-9;:TRIGGER:A:RUNT:SOURCE CH1;POLARITY POSITIVE;WHEN
OCCURS;WIDTH 4.0000E-9;:TRIGGER:A:TRANSITION:SOURCE
CH1;POLARITY POSITIVE;WHEN SLOWER;DELTATIME
4.0000E-9;:TRIGGER:A:VIDEO :POLARITY POSITIVE;SOURCE
CH1;STANDARD NTSC;SYNC ALLLINES;HOLDOFF:FIELD 0.0000;:
TRIGGER:A:VIDEO:CUSTOM:FORMAT PROGRESSIVE;SCAN
RATE15K;:TRIGGER:A:VIDEO:LINE 1;H DTV:FORMAT
HD1080I60;:TRIGGER:A:BUS:SOURCE B1;B1:I2C:CONDITION
START;DATA:VALUE "XXXXXXXX";SIZE 1;START 0.0000;DIRECTION
NOCARE;:TRIGGER:A:BUS:B1:I2C:ADDRESS:MODE
ADDR7;TYPE USER;VALUE
"XXXXXXXX";:TRIGGER:A:BUS:B1:SPI:CONDITION MOSI;DATA:OUT
:VALUE "XXXXXXXX";:TRIGGER:A:BUS:B1:SPI:DATA:IN:VALUE
"XXXXXXXX";:TRIGGER:A:BUS: B1:SPI:DATA:SIZE 1;START 0.0000;:

```

## TRIGger:A:BUS

Sets or returns the trigger type: I2C, CAN, SPI, and RS232. There are up to two serial buses, B1, B2, depending on your instrument model. Each can be independently set to one of the serial trigger types. The serial parameters related to the trigger are broken into two sections: Trigger:A:SERIAL xxx, consisting of parameters the user will change frequently, and BUS:B1:xxx, consisting of parameters the user will specify once (bus definition).

**Conditions** Requires a DPO2AUTO, DPO2EMBD, or DPO2COMP application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS {I2C|SPI|CAN|RS232}  
TRIGger:A:BUS?

**Arguments** I2C specifies the Inter-IC bus.  
SPI specifies the Serial Peripheral Interface bus.  
CAN specifies the Controller Area Network bus.

## TRIGger:A:BUS:B<x>:CAN:CONDition

Sets or returns the CAN trigger condition for bus <x>, where x is the bus number.

**Conditions** Requires a DPO2AUTO application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:CAN:CONDition  
{SOF|FRAMEtype|IDentifier|DATA|IDANDDATA|EOF|ACKMISS}  
TRIGger:A:BUS:B<x>:CAN:CONDition?

**Arguments** SOF enables triggering on the start of frame.  
FRAMEtype enables triggering on the type of frame.  
IDentifier enables triggering on a matching identifier.  
DATA enables triggering on matching data.  
IDANDDATA enables triggering on a matching identifier and matching data.  
EOF enables triggering on the end of frame.  
ACKMISS enables triggering on a missing acknowledge.

**Examples** TRIGGER:A:BUS:B1:CAN:CONDITION? might return :TRIGGER:A:BUS:B1:CAN:CONDITION EOF indicating an end of file condition.  
TRIGGER:A:BUS:B1:CAN:CONDITION DATA enables triggering on matching CAN data.

## TRIGger:A:BUS:B<x>:CAN:DATA:DIRection

Sets or returns the CAN trigger data direction to be valid on a Read, Write, or Either condition for bus <x>, where x is the bus number. This applies only, if the trigger condition is ID.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | Requires a DPO2AUTO application module.   |
| <b>Group</b>      | Trigger   |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:CAN:DATA:DIRection {READ WRITE NOCARE}<br>TRIGger:A:BUS:B<x>:CAN:DATA:DIRection?                                       |
| <b>Arguments</b>  | READ sets the CAN data direction to READ.<br>WRITE sets the CAN data direction to WRITE.<br>NOCARE sets the CAN data direction to either. |
| <b>Examples</b>   | TRIGGER:A:BUS:B1:CAN:DATA:DIRECTION WRITE sets the CAN data direction to Write.   |

## TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier

Sets or returns the CAN data qualifier for bus <x>, where x is the bus number. This applies only, if the trigger condition is IDANDDATA or DATA.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | Requires a DPO2AUTO application module.   |
| <b>Group</b>      | Trigger   |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier<br>{LESSThan MOREThan EQUa1 UNEQua1 LESSEQua1 MOREEQua1}<br>TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier?  |
| <b>Arguments</b>  | LESSThan sets the oscilloscope to trigger when the data is less than the qualifier value.<br>MOREThan sets the oscilloscope to trigger when the data is more than the qualifier value.<br>EQUa1 sets the oscilloscope to trigger when the data is equal to the qualifier value. |

UNEQUAL sets the oscilloscope to trigger when the data is not equal to the qualifier value.

LESSEQUAL sets the oscilloscope to trigger when the data is less than or equal to the qualifier value.

MOREEQUAL sets the oscilloscope to trigger when the data is more than or equal to the qualifier value.

**Examples** TRIGGER:A:BUS:B1:CAN:DATA:QUALIFIER LESSTHAN sets the oscilloscope to trigger when the data is less than the qualifier value.

TRIGGER:A:BUS:B1:CAN:DATA:QUALIFIER? might return  
:TRIGGER:A:BUS:B1:CAN:DATA:QUALIFIER THAN, indicating that the oscilloscope is set to trigger when the data is than the qualifier value.

## TRIGger:A:BUS:B<x>:CAN:DATA:SIZE

Sets or returns the length of the data string in bytes for a CAN trigger if the condition is DATA or IDANDDATA. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:CAN:DATA:SIZE <NR1>  
TRIGger:A:BUS:B<x>:CAN:DATA:SIZE?

**Arguments** <NR1> is the length of the data string in bytes.

## TRIGger:A:BUS:B<x>:CAN:DATA:VALue

Sets or returns the binary data string to be used for a CAN trigger if the trigger condition is ID or IDANDDATA. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:CAN:DATA:VALue <QString>  
TRIGger:A:BUS:B<x>:CAN:DATA:VALue?

**Arguments** <QString> is the CAN data value in binary format. The only allowed characters in the QString are 0, 1, and X.

**Examples** TRIGGER:A:BUS:B1:CAN:DATA:VALUE 1011 sets the CAN data value to 1011.

## TRIGger:A:BUS:B<x>:CAN:FRAMEType

Sets or returns the frame type for a CAN FRAMEType trigger. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2AUTO application module. This command is only valid when the [TRIGger:A:BUS:B<x>:CAN:CONDition](#) is FRAMETYPE.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:CAN:FRAMEType  
{DATA|REMOte|ERRor|OVERLoAd}  
TRIGger:A:BUS:B<x>:CAN:FRAMEType?

**Arguments** DATA specifies a data frame type.  
REMOte specifies a remote frame type.  
ERRor specifies an error frame type.  
OVERLoAd specifies an overload frame type.

**Examples** TRIGGER:A:BUS:B1:CAN:FRAMETYPE DATA sets the CAN trigger frame type to DATA.

## TRIGger:A:BUS:B<x>:CAN{:IDentifier|:ADDReSS}:MODE

Sets or returns the CAN addressing mode for bus <x>, where x is the bus number. Use this command to do the following:

- Trigger on ID
- Trigger in IDANDDATA



|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.   |
| <b>Group</b>      | Trigger  |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:CAN{:IDentifier :ADDRESS}:MODE<br>{STandard EXTended}<br>TRIGger:A:BUS:B<x>:CAN{:IDentifier :ADDRESS}:MODE? |
| <b>Arguments</b>  | STandard specifies the standard addressing mode.<br>EXTended specifies the extended addressing mode.                           |

### TRIGger:A:BUS:B<x>:CAN{:IDentifier|:ADDRESS}:VALue

Sets or returns the binary address string used for a CAN trigger if the trigger condition is ID or IDANDDATA. Applies to bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.   |
| <b>Group</b>      | Trigger  |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:CAN{:IDentifier :ADDRESS}:VALue <QString><br>TRIGger:A:BUS:B<x>:CAN{:IDentifier :ADDRESS}:VALue?                |
| <b>Arguments</b>  | <QString> is up to 29 bits specifying the binary CAN identifier value. The only allowed characters in the QString are 0, 1, and X. |
| <b>Examples</b>   | TRIGGER:A:BUS:B1:CAN:IDENTIFIER:VALUE 1011 sets the CAN trigger identifier value to 1011.  |

### TRIGger:A:BUS:B<x>:I2C:ADDRESS:MODE

Sets or returns the I2C address mode to 7 or 10-bit. Applies to bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module. |
| <b>Group</b>      | Trigger  |

**Syntax** TRIGger:A:BUS:B<x>:I2C:ADDRESS:MODE {ADDR7|ADDR10}  
 TRIGger:A:BUS:B<x>:I2C:ADDRESS:MODE?

**Arguments** ADDR7 specifies the 7-bit I<sup>2</sup>C address mode.  
 ADDR10 specifies the 10-bit I<sup>2</sup>C address mode.

**Examples** TRIGGER:A:BUS:B1:I2C:ADDRESS:MODE ADDR10 sets the I2C address mode to 10-bit.

## TRIGger:A:BUS:B<x>:I2C:ADDRESS:TYPE

Sets or returns the I<sup>2</sup>C address type. The only supported address type is USER.  
 Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:I2C:ADDRESS:TYPE  
 {GENERALCALL|STARTBYTE|HSMODE|EEPROM|USER}  
 TRIGger:A:BUS:B<x>:I2C:ADDRESS:TYPE?

**Arguments** GENERALCALL specifies a general call address.  
 STARTBYTE specifies a start byte address.  
 HSMODE specifies a high-speed mode address.  
 EEPROM specifies an EEPROM address.  
 USER specifies a user address.

## TRIGger:A:BUS:B<x>:I2C:ADDRESS:VALUE

Sets or returns the binary address string used for the I<sup>2</sup>C trigger if the trigger condition is ADDRESS or ADDRANDDATA. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:I2C:ADDRESS:VALue <QString>  
TRIGger:A:BUS:B<x>:I2C:ADDRESS:VALue?

**Arguments** <QString> is up to 7 or 10-bits depending on the address mode that specifies the address. The only allowed characters in the QString are 0, 1, and X.

**Examples** TRIGGER:A:BUS:B1:I2C:ADDRESS:VALUE 1011 sets the I<sup>2</sup>C address value to XXX1011.

## TRIGger:A:BUS:B<x>:I2C:CONDition

Sets or returns the trigger condition for an I<sup>2</sup>C trigger. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:I2C:CONDition  
{START|STOP|REPEATstart|ACKMISS|ADDRESS|DATA|ADDRANDDATA}  
TRIGger:A:BUS:B<x>:I2C:CONDition?

**Arguments** START specifies a search based on start condition.  
STOP specifies a search based on stop condition.  
REPEATstart specifies a search based on repeat of start condition.  
ACKMISS specifies a search based on missing acknowledgement condition.  
ADDRESS specifies a search based on address.  
DATA specifies a search based on data.  
ADDRANDDATA specifies a search based on address and data.

**Examples** TRIGGER:A:BUS:B1:I2C:CONDITION START specifies start as the I<sup>2</sup>C trigger condition.

## TRIGger:A:BUS:B<x>:I2C:DATA:DIRection

Sets or returns the I<sup>2</sup>C trigger type to be valid on a Read, Write, or Either condition. Read or write is indicated by the R/W bit in the I<sup>2</sup>C protocol. Applies to bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.   |
| <b>Group</b>      | Trigger  |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:I2C:DATA:DIRection {READ WRITE NOCARE}<br>TRIGger:A:BUS:B<x>:I2C:DATA:DIRection?  |
| <b>Arguments</b>  | READ specifies read as the data direction.<br>WRITE specifies write as the data direction.<br>NOCARE specifies either as the data direction. |
| <b>Examples</b>   | TRIGGER:A:BUS:B1:I2C:DATA:DIRECTION WRITE specifies write as the I <sup>2</sup> C data direction.  |

## TRIGger:A:BUS:B<x>:I2C:DATA:SIZE

Sets or returns the length of the data string in bytes to be used for an I<sup>2</sup>C trigger if the trigger condition is DATA or ADDRANDDATA. Applies to bus <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.                        |
| <b>Group</b>      | Trigger   |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:I2C:DATA:SIZE <NR1><br>TRIGger:A:BUS:B<x>:I2C:DATA:SIZE? |
| <b>Arguments</b>  | <NR1> is the length of the data string in bytes.                            |

## TRIGger:A:BUS:B<x>:I2C:DATA:VALue

Sets or returns the binary data string used for I2C triggering if the trigger condition is DATA or ADDRANDDATA. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2EMBD application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:I2C:DATA:VALue <QString>  
TRIGger:A:BUS:B<x>:I2C:DATA:VALue?

**Arguments** <QString> is the binary data string, where the number of bits is 8 times the number of bytes specified. The only allowed characters in the string are 0, 1, and X.

## TRIGger:A:BUS:B<x>:LIN:CONDition

Sets or returns the trigger condition for LIN.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** TRIGger:A:BUS:B<x>:LIN:CONDition  
{SYNCFieLd|IDentifier|DATA|IDANDDATA|WAKEup|SLEEP|ERROR}  
TRIGger:A:BUS:B<x>:LIN:CONDition?

**Arguments** SYNCFieLd sets the LIN trigger condition to sync field.  
IDentifier sets the LIN trigger condition to identifier.  
DATA sets the LIN trigger condition to data.  
IDANDDATA sets the LIN trigger condition to ID and data.  
WAKEup sets the LIN trigger condition to wake up.  
SLEEP sets the LIN trigger condition to sleep.  
ERROR sets the LIN trigger condition to error.

**Examples** TRIGGER:A:BUS:B1:LIN:CONDITION ERROR sets the LIN trigger condition to error.

TRIGGER:A:BUS:B1:LIN:CONDITION? might return  
 TRIGGER:A:BUS:B1:LIN:CONDITION SYNCFIELD indicating that the LIN trigger condition is sync field.

## TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue

Sets or returns the binary data string to be used for LIN trigger if trigger condition is ID or IDANDDATA.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue <QString>  
 TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue?

**Arguments** <QString> is a quoted string that is the binary data string used for LIN trigger if the trigger condition is ID or IDANDDATA.

**Examples** TRIGGER:A:BUS:B1:LIN:DATA:HIVALUE "11001010" sets the high value to 11001010.

TRIGGER:A:BUS:B1:LIN:DATA:HIVALUE? might return  
 TRIGGER:A:BUS:B1:LIN:DATA:HIVALUE "XXXXXXXX" indicating that the high value is don't care.

## TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier

Sets or returns the LIN data qualifier. This only applies if the trigger condition is IDANDDATA or DATA.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier  
 {LESSThan|MOREThan|EQUal|UNEQUal|LESSEQUal|MORREEQUal|INrange|OUTrange}  
 TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier?

**Arguments**    `LESSThan` sets the LIN data qualifier to less than.  
                   `MOREThan` sets the LIN data qualifier to greater than.  
                   `EQUa1` sets the LIN data qualifier to equal.  
                   `UNEQUa1` sets the LIN data qualifier to not equal.  
                   `LESSEQUa1` sets the LIN data qualifier to less than or equal.  
                   `MOREEQUa1` sets the LIN data qualifier to greater than or equal.  
                   `INrange` sets the LIN data qualifier to in range.  
                   `OUTrange` sets the LIN data qualifier to out of range.

**Examples**     `TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER OUTRANGE` sets the data qualifier to out of range.  
                   `TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER?` might return  
                   `TRIGGER:A:BUS:B1:LIN:DATA:QUALIFIER EQUAL` indicating that the data qualifier is set to equal.

## TRIGger:A:BUS:B<x>:LIN:DATA:SIZE

Sets or returns the length of the data string in bytes to be used for LIN trigger.

**Conditions**    This command requires a DPO2AUTO application module.

**Group**         Bus

**Syntax**        `TRIGger:A:BUS:B<x>:LIN:DATA:SIZE <NR1>`  
                   `TRIGger:A:BUS:B<x>:LIN:DATA:SIZE?`

**Arguments**    `<NR1>` is the size of the data string in bytes.

**Examples**     `TRIGGER:A:BUS:B1:LIN:DATA:SIZE 8` sets the data size to 8 bytes.  
                   `TRIGGER:A:BUS:B1:LIN:DATA:SIZE?` might return  
                   `TRIGGER:A:BUS:B1:LIN:DATA:SIZE 1` indicating that the data size is 1 byte.

## TRIGger:A:BUS:B<x>:LIN:DATA:VALue

Sets or returns the binary data string to be used for LIN trigger condition if trigger condition is ID or IDANDDATA.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.  |
| <b>Group</b>      | Bus   |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:LIN:DATA:VALUE <QString><br>TRIGger:A:BUS:B<x>:LIN:DATA:VALUE?   |
| <b>Arguments</b>  | <QString> is a quoted string that is the LIN trigger data value.  |
| <b>Examples</b>   | TRIGGER:A:BUS:B1:LIN:DATA:VALUE "11001101" sets the data value to 11001101.<br><br>TRIGGER:A:BUS:B1:LIN:DATA:VALUE? might return TRIGGER:A:BUS:B1:LIN:DATA:VALUE "XXXXXXXX" indicating that the data value is don't care. |

## TRIGger:A:BUS:B<x>:LIN:ERRTYPE

Sets or returns the error type be used for LIN trigger.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO application module.   |
| <b>Group</b>      | Bus  |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:LIN:ERRTYPE {SYNC PARity CHECKsum}<br>TRIGger:A:BUS:B<x>:LIN:ERRTYPE?   |
| <b>Arguments</b>  | SYNC sets the LIN error type to SYNC.<br><br>PARity sets the LIN error type to parity.<br><br>CHECKsum sets the LIN error type to checksum.  |
| <b>Examples</b>   | TRIGGER:A:BUS:B1:LIN:ERRTYPE CHECKSUM sets the LIN error type to checksum.<br><br>TRIGGER:A:BUS:B1:LIN:ERRTYPE? might return TRIGGER:A:BUS:B1:LIN:ERRTYPE SYNC indicating that the LIN error type is SYNC. |



**TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue**

Sets or returns the binary address string used for LIN trigger if the trigger condition is ID or IDANDDATA.

**Conditions** This command requires a DPO2AUTO application module.

**Group** Bus

**Syntax** TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue <QString>  
TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue?

**Arguments** <QString> is the binary address string used for LIN trigger if the trigger condition is ID or IDANDDATA.

**Examples** TRIGGER:A:BUS:B1:LIN:IDENTIFIER:VALUE "110010" sets the identifier value to 110010.  
  
TRIGGER:A:BUS:B1:LIN:IDENTIFIER:VALUE? might return  
TRIGGER:A:BUS:B1:LIN:IDENTIFIER:VALUE "XXXXXX" indicating that the identifier value is XXXXXX.

**TRIGger:A:BUS:B<x>:PARAllel:VALue**

Sets or returns the binary data string to be used for a parallel trigger. Applies to bus <x>, where x is the bus number.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:PARAllel:VALue <QString>  
TRIGger:A:BUS:B<x>:PARAllel:VALue?

**Arguments** <QString> is the binary data string.

**TRIGger:A:BUS:B<x>:RS232C:CONDition**

Sets or returns the condition for a RS232C trigger, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

|                  |  |
|------------------|--|
| <b>Group</b>     | Trigger  |
| <b>Syntax</b>    | <pre>TRIGger:A:BUS:B&lt;x&gt;:RS232C:CONDition {RXSTArt RXDATA RXENDPacket RXPARity TXSTArt  TXDATA TXENDPacket TXPARity } TRIGger:A:BUS:B&lt;x&gt;:RS232C:CONDition?</pre>  |
| <b>Arguments</b> | <p>RXSTArt specifies a search based on the RX Start Bit.</p> <p>RXDATA specifies a search based on RX Data.</p> <p>RXENDPacket specifies a search based on the RX End of Packet condition.</p> <p>RXPARity specifies a search based on the RX parity.</p> <p>TXSTArt specifies a search base on the TX Start Bit.</p> <p>TXDATA specifies a search based on TX Data.</p> <p>TXENDPacket specifies a search based on the TX End of Packet condition.</p> <p>TXPARity specifies a search based on the TX parity.</p> |

## TRIGger:A:BUS:B<x>:RS232C:RX:DATA:SIZE

Sets or returns the length of the data string in Bytes for a RS232 Trigger if the trigger condition is RXDATA. Applies to bus <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2COMP application module.  |
| <b>Group</b>      | Trigger   |
| <b>Syntax</b>     | <pre>TRIGger:A:BUS:B&lt;x&gt;:RS232C:RX:DATA:SIZE &lt;NR1&gt; TRIGger:A:BUS:B&lt;x&gt;:RS232C:RX:DATA:SIZE?</pre> |
| <b>Arguments</b>  | <NR1> is the length of the data string in bytes.  |

## TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue

Sets or returns the binary data string for a RS232 trigger if the trigger condition involves RX. Applies to bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2COMP application module. |
|-------------------|--|

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue  
TRIGger:A:BUS:B<x>:RS232C:RX:DATA:VALue?

**Arguments** <Qstring> is the binary data string to be used for the trigger.

## TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE

Sets or returns the length of the data string for a RS232 trigger if the trigger condition is TXDATA. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE <NR1>  
TRIGger:A:BUS:B<x>:RS232C:TX:DATA:SIZE?

**Arguments** <NR1> is the length of the data string in Bytes.

## TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue

Sets or returns the binary data string for a RS232 trigger if the condition involves TX. Applies to bus <x>, where x is the bus number.

**Conditions** This command requires a DPO2COMP application module.

**Group** Trigger

**Syntax** TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue  
TRIGger:A:BUS:B<x>:RS232C:TX:DATA:VALue?

**Arguments** <Qstring> is the binary data string to be used for the trigger.

## TRIGger:A:BUS:B<x>:SPI:CONDition

Sets or returns the trigger condition for a SPI trigger. Applies to bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.   |
| <b>Group</b>      | Trigger  |
| <b>Syntax</b>     | <pre>TRIGger:A:BUS:B&lt;x&gt;:SPI:CONDition {SS STARTofframe MISO MOSI MISOMOSI} TRIGger:A:BUS:B&lt;x&gt;:SPI:CONDition?</pre>   |
| <b>Arguments</b>  | <p>SS specifies the Slave Selection condition.</p> <p>STARTofframe specifies the start of frame condition.</p> <p>MISO specifies the Master-In Slave-Out condition.</p> <p>MOSI specifies the Master-Out Slave-In condition.</p> <p>MISOMOSI specifies the Master-In Slave-Out and Master-Out Slave-In conditions.</p> |

## TRIGger:A:BUS:B<x>:SPI:DATA{:IN|:MISO}:VALue

Sets or returns the binary data string to be used for a SPI trigger if the trigger condition is MISO or MISOMOSI. Applies to bus <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.  |
| <b>Group</b>      | Trigger   |
| <b>Syntax</b>     | <pre>TRIGger:A:BUS:B&lt;x&gt;:SPI:DATA{:IN :MISO}:VALue &lt;QString&gt; TRIGger:A:BUS:B&lt;x&gt;:SPI:DATA{:IN :MISO}:VALue?</pre>   |
| <b>Arguments</b>  | <p>&lt;QString&gt; is the binary data string, where the number of bits is 8 times the number of bytes specified. The only allowed characters in the string are 0, 1, and X.</p> |

## TRIGger:A:BUS:B<x>:SPI:DATA{:OUT|:MOSI}:VALue

Sets or returns the binary data string to be used for a SPI trigger if the trigger condition is MOSI or MISOMOSI. Applies to bus <x>, where x is the bus number.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.   |
| <b>Group</b>      | Trigger  |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:SPI:DATA{:OUT :MOSI}:VALue <QString><br>TRIGger:A:BUS:B<x>:SPI:DATA{:OUT :MOSI}:VALue?  |
| <b>Arguments</b>  | <QString> is the binary data string with the number of bits specified by the <a href="#">TRIGger:A:BUS:B&lt;x&gt;:SPI:DATA:SIZE</a> command. The only allowed characters in the QString are 0, 1, and X. |

## TRIGger:A:BUS:B<x>:SPI:DATA:SIZE

Sets or returns the length of the data string to be used for a SPI trigger if the trigger condition is MISO, MOSI, or MISOMOSI. Applies to bus <x>, where x is the bus number.

|                   |   |
|-------------------|---|
| <b>Conditions</b> | This command requires a DPO2EMBD application module.                        |
| <b>Group</b>      | Trigger   |
| <b>Syntax</b>     | TRIGger:A:BUS:B<x>:SPI:DATA:SIZE <NR1><br>TRIGger:A:BUS:B<x>:SPI:DATA:SIZE? |
| <b>Arguments</b>  | <NR1> is the length of the data string in bytes.                            |

## TRIGger:A:BUS:SOURce

Sets or returns the source for a Serial bus trigger.

|                   |  |
|-------------------|--|
| <b>Conditions</b> | This command requires a DPO2AUTO or DPO2EMBD application module. |
| <b>Group</b>      | Trigger  |

**Syntax** TRIGger:A:BUS:SOURce {B1|B2}  
TRIGger:A:BUS:SOURce?

**Arguments** B1 specifies the Bus 1 source.  
B2 specifies the Bus 2 source.

## TRIGger:A:EDGE? (Query Only)

Returns the trigger source, coupling, and slope for the A edge trigger.

**Group** Trigger

**Syntax** TRIGger:A:EDGE?

**Related Commands** [TRIGger:A:PULse?](#), [TRIGger:A:LOGic?](#)

**Examples** TRIGGER:A:EDGE? might return :TRIGGER:A:EDGE:SOURCE CH1;COUPLING DC; SLOPE RISE indicating that the trigger source, coupling, and slope for the A edge trigger.

## TRIGger:A:EDGE:COUPLing

Sets or returns the type of coupling for the A edge trigger.

**Group** Trigger

**Syntax** TRIGger:A:EDGE:COUPLing {DC|HFRej|LFRej|NOISerej}  
TRIGger:A:EDGE:COUPLing?

**Related Commands** [TRIGger:A:EDGE:SOURce](#), [TRIGger:A:EDGE:SLOpe](#)

**Arguments** DC specifies DC trigger coupling, which passes all input signals to the trigger circuitry.

HFRej specifies high-frequency rejection coupling, which attenuates signals above 50 kHz before passing the signals to the trigger circuitry.

LFRej specifies low-frequency rejection coupling, which attenuates signals below 50 kHz before passing the signals to the trigger circuitry.

`NOISErej` specifies noise-rejection coupling, which provides stable triggering by increasing the trigger hysteresis. Increased hysteresis reduces the trigger sensitivity to noise but may require greater trigger signal amplitude.

**Examples** `TRIGGER:A:EDGE:COUPLING DC` sets the A edge trigger coupling to DC.  
`TRIGGER:A:EDGE:COUPLING?` might return `:TRIGGER:A:EDGE:COUPLING DC` indicating that the A edge trigger coupling is set to DC.

## TRIGger:A:EDGE:SLOpe

Sets or returns the slope for the A edge trigger.

**Group** Trigger

**Syntax** `TRIGger:A:EDGE:SLOpe {RISe|FALL}`  
`TRIGger:A:EDGE:SLOpe?`

**Related Commands** [TRIGger:A:EDGE:SOUrce](#), [TRIGger:A:EDGE:COUPling](#)

**Arguments** `RISe` specifies to trigger on the rising or positive edge of a signal.  
`FALL` specifies to trigger on the falling or negative edge of a signal.

**Examples** `TRIGGER:A:EDGE:SLOPE RISE` sets the A edge trigger slope to positive, which triggers on the rising edge of the signal.  
`TRIGGER:A:EDGE:SLOPE?` might return `:TRIGGER:A:EDGE:SLOPE FALL` indicating that the A edge trigger slope is negative.

## TRIGger:A:EDGE:SOUrce

Sets or returns the source for the A edge trigger.

**Group** Trigger

**Syntax** `TRIGger:A:EDGE:SOUrce {{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15|EXT|LINE|AUX}}`  
`TRIGger:A:EDGE:SOUrce?`

**Related Commands** [TRIGger:A:EDGE:SLOpe](#), [TRIGger:A:EDGE:COUPling](#)

**Arguments** CH1–CH4 specifies an analog input channel as the A edge trigger source.  
D0–D15 specifies a digital input channel as the A edge trigger source.  
EXT specifies an external trigger using the Aux In connector located on the front panel of the oscilloscope.  
LINE specifies the AC line as the trigger source.  
AUX specifies the Auxiliary Input as the trigger source (if available on your oscilloscope).

**Examples** TRIGGER:A:EDGE:SOURCE CH1 sets channel 1 as the A edge trigger source.  
TRIGGER:A:EDGE:SOURCE? might return :TRIGGER:A:EDGE:SOURCE CH1 indicating that channel 1 is the A edge trigger source.

## TRIGger:A:HOLDoff? (Query Only)

Returns the A trigger holdoff parameters. These parameters specify the time period during which the trigger circuitry is not looking to generate a trigger event.

**Group** Trigger

**Syntax** TRIGger:A:HOLDoff?

**Related Commands** [TRIGger:A:HOLDoff:TIME](#)

**Examples** TRIGGER:A:HOLDOFF? might return :TRIGGER:A:HOLDOFF:TIME 900.0000E-09;BY DEFAULT indicating that the A edge trigger holdoff time (by default) is 900 ns.

## TRIGger:A:HOLDoff:TIME

Sets or returns the A trigger holdoff time.

**Group** Trigger

**Syntax** TRIGger:A:HOLDoff:TIME <NR3>  
TRIGger:A:HOLDoff:TIME?



**Arguments** <NR3> specifies the holdoff time in seconds. The range is from 20 ns through 8.0 s.

**Examples** TRIGGER:A:HOLDOFF:TIME ? might return :TRIGGER:A:HOLDOFFTIME 1.2000E-06 indicating that the A trigger holdoff time is set to 1.2  $\mu$ s.  
TRIGGER:A:HOLDOFF:TIME 10 sets the A trigger holdoff time to 10 s.

## TRIGger:A:LEVel

Sets or returns the trigger level for the A trigger.

**Group** Trigger

**Syntax** TRIGger:A:LEVel {ECL|TTL|<NR3>}  
TRIGger:A:LEVel?

**Arguments** ECL specifies a preset ECL high level of -1.3V.  
TTL specifies a preset TTL high level of 1.4V.  
<NR3> specifies the trigger level in user units (usually volts).

**Examples** TRIGGER:A:LEVEL? might return :TRIGGER:A:LEVEL 1.3000E+00 indicating that the A edge trigger is set to 1.3 V.  
TRIGGER:A:LEVEL TTL sets the A edge trigger to TTL high level, which is 1.4 V.

## TRIGger:A:LEVel:AUXin

Sets or returns the trigger level for the AUXIN port.

**Group** Trigger

**Syntax** TRIGger:A:LEVel:AUXin {<NR3>|ECL|TTL}  
TRIGger:A:LEVel:AUXin?

**Arguments** <NR3> specifies the trigger level, in volts.  
ECL specifies a preset ECL trigger level of -1.3 V.  
TTL specifies a preset TTL trigger level of 1.4 V.

- Examples** TRIGGER:A:LEVEL:AUXIN ECL sets the auxiliary input trigger level to -1.3 volts.
- TRIGGER:A:LEVEL:AUXIN? might return TRIGGER:A:LEVEL:AUXIN 0.0E+0 indicating that the auxiliary input trigger level is 0.0 volts.

## TRIGger:A:LEVel:CH<x>

Sets or returns the trigger level for the specified channel. Each channel can have an independent level.

**Group** Trigger

**Syntax** TRIGger:A:LEVel:CH<x> {<NR3>|TTL|ECL}  
TRIGger:A:LEVel:CH<x>?

**Arguments** <NR3> specifies the trigger level in user units (usually volts).

TTL specifies a preset TTL high level of 1.4 V.

ECL specifies a preset ECL high level of -1.3 V.

- Examples** TRIGGER:A:LEVEL:CH2? might return :TRIGGER:A:LEVEL:CH2 1.3000E+00 indicating that the A edge trigger is set to 1.3 V for channel 2.
- TRIGGER:A:LEVEL:CH3 TTL sets the A edge trigger to TTL high level for channel 3.

## TRIGger:A:LEVel:D<x>

Sets or returns the trigger level for the specified digital channel <x>, where x is the channel number. The digital channels do not have independent trigger levels. Channels D0–D7 share one common trigger level and D8–D15 share another common trigger level. For example: Changing the trigger level for D3 changes the levels for D0–D7.

**Group** Trigger

**Syntax** TRIGger:A:LEVel:D<x> {<NR3>|ECL|TTL}  
TRIGger:A:LEVel:D<x>?

**Arguments** ECL specifies a preset ECL high level of -1.3 V.  
 TTL specifies a preset TTL high level of 1.4 V.  
 <NR3> specifies the trigger level in volts.

## TRIGger:A:LOGIc? (Query Only)

Returns all of the A logic trigger parameters.

**Group** Trigger

**Syntax** TRIGger:A:LOGIc?

**Related Commands** [TRIGger:A:LOGIc:CLAss](#)

**Examples** TRIGGER:A:LOGIC? might return :TRIGGER:A:LOGIC:CLASS  
 SETHOLD;FUNCTION AND;THRESHOLD:CH1 20.0000E-3;CH2 0.0000;  
 CH3 0.0000;CH4 0.0000;:TRIGGER:A:LOGIC:INPUT:CH1  
 X;CH2 X;CH3 X;CH4 X;CLOCK:SOURCE NONE;EDGE  
 RISE;:TRIGGER:A:LOGIC:PATTERN:INPUT:CH1 X;CH2 X;CH3  
 X;CH4 X;:TRIGGER :A:LOGIC:PATTERN:WHEN TRUE;WHEN:LESSLIMIT  
 4.0000E-9;LIMIT 4.0000E-9;:TRIGGER:A:LOGIC:PATTERN:DELTATIME  
 4.0000E-9

## TRIGger:A:LOGIc:CLAss

Sets or returns the class of the Logic Trigger. This command is used in conjunction with the [TRIGger:A:TYPE](#) command.

**Group** Trigger

**Syntax** TRIGger:A:LOGIc:CLAss {LOGIC|SETHo]d}  
 TRIGger:A:LOGIc:CLAss?

**Related Commands** [TRIGger:A:TYPE](#), [TRIGger:A:PULSe:CLAss](#)

**Arguments** LOGIC sets the oscilloscope to trigger on logical combinations of the channels.  
 When the [TRIGger:A:LOGIc:INPut:CLOCK:SOURce](#) is NONE, LOGIC sets the oscilloscope to trigger when the specified logical combinations of channels 1, 2,

3, and 4 are met on four-channel oscilloscopes. On two-channel oscilloscopes, only channel 1 and channel 2 are available.

When the [TRIGger:A:LOGic:INPut:CLOCK:SOUrce](#) is set to one of the channels, LOGIC sets the oscilloscope to trigger when the specified logical combinations of the remaining channels is true during a transition on the clock channel.

SETHoId sets the oscilloscope to trigger on setup and hold violations between a data source and a clock source. Use one channel input as the clock signal and a second channel input as the data input. The clocking and data levels are used to determine if a clock or data transition has occurred.

**Examples**     `TRIGGER:A:LOGIC:CLASS?` might return `:TRIGGER:A:LOGIC:CLASS LOGIC`  
`TRIGGER:A:LOGIC:CLASS LOGIC` sets the trigger A logic class to LOGIC, which causes the oscilloscope to trigger when the specified logical combinations of channels 1, 2, 3, and 4 are met.

## TRIGger:A:LOGic:FUNction

Sets or returns the logical combination of the input channels for the A pattern and A state logic triggers.

**Group**     Trigger

**Syntax**     `TRIGger:A:LOGic:FUNction {AND|NAND}`  
`TRIGger:A:LOGic:FUNction?`

**Related Commands**     [TRIGger:A:LOGic:INPut:CH<x>](#)

**Arguments**     AND specifies to trigger if all conditions are true.  
 NAND specifies to trigger if any of the conditions is false.

**Examples**     `TRIGGER:A:LOGIC:FUNCTION?` might return `:TRIGGER:A:LOGIC:FUNCTION NAND` which indicates that the oscilloscope will trigger if the AND logic conditions are false.  
`TRIGGER:A:LOGIC:FUNCTION AND` sets the logical combination of channels to be true when all conditions are true.

## TRIGger:A:LOGic:INPut? (Query Only)

Returns the logic input values for all channels. If a clock channel is defined, it returns the clock source and edge.

**Group** Trigger

**Syntax** TRIGger:A:LOGic:INPut?

**Examples** TRIGGER:A:LOGIC:INPUT? might return :TRIGGER:A:LOGIC:INPUT:CH1 HIGH;CH2 X;CH3 X indicating that a logic high is expected on channel 1 while channel 2 and channel three are “don’t care.”

## TRIGger:A:LOGic:INPut:CH<x>

Sets or returns the logical input condition for the channel specified by <x>.

**Group** Trigger

**Syntax** TRIGger:A:LOGic:INPut:CH<x> {HIGH|LOW|X}  
TRIGger:A:LOGic:INPut:CH<x>?

**Arguments** HIGH specifies the logic high.  
LOW specifies the logic low.  
X specifies a "don't care" state.

**Examples** TRIGGER:A:LOGIC:INPUT:CH1? might return :TRIGGER:LOGIC:INPUT:CH1 X indicating that the setting for the A logic trigger input to channel 1 does not matter.  
TRIGGER:A:LOGIC:INPUT:CH2 HIGH sets the A logic trigger input to logic HIGH for channel 2.

## TRIGger:A:LOGic:INPut:CLOCK:EDGE

Sets the polarity of the clock channel.

**Group** Trigger

**Syntax** TRIGger:A:LOGIC:INPut:CLOCK:EDGE {FALL|RISE|EITHer}  
 TRIGger:A:LOGIC:INPut:CLOCK:EDGE?

**Arguments** RISE specifies to trigger on the rising or positive edge of a signal.  
 FALL specifies to trigger on the falling or negative edge of a signal.  
 EITHer specifies to trigger either on the falling or rising edge of a signal.

## TRIGger:A:LOGIC:INPut:CLOCK:SOURce

Sets or returns the channel to use as the clock source. The clock can be selected as NONE. A selection of None implies pattern trigger. Any other selection implies state trigger.

**Group** Trigger

**Syntax** TRIGger:A:LOGIC:INPut:CLOCK:SOURce {CH1|CH2|CH3|CH4|  
 D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15|NONE}  
 TRIGger:A:LOGIC:INPut:CLOCK:SOURce?

**Arguments** CH1–CH4 specifies the analog input channel source.  
 D0–D15 specifies the digital input channel source.  
 NONE specifies a Pattern trigger.

## TRIGger:A:LOGIC:INPut:D<x>

Sets or returns the logic pattern for a trigger on digital channel <x>, where x is the channel number.

**Group** Trigger

**Syntax** TRIGger:A:LOGIC:INPut:D<x> {HIGH|LOW|X}  
 TRIGger:A:LOGIC:INPut:D<x>?

**Arguments** High specifies the logic high state.  
 Low specifies the logic low state.  
 X specifies a "don't care" state.

## TRIGger:A:LOGic:PATtern? (Query Only)

Returns the conditions used for generating an A logic pattern trigger, with respect to the defined input pattern, and identifies the time that the selected pattern may be true and still generate the trigger.

**Group** Trigger

**Syntax** TRIGger:A:LOGIC:PATtern?

**Examples** TRIGGER:A:LOGIC:PATTERN? might return  
 :TRIGGER:A:LOGIC:PATTERN:INPUT:CH1 HIGH;CH2  
 LOW;CH3 X;CH4 X;:TRIGGER:A:LOGIC:PATTERN:WHEN  
 LESSTHAN;WHEN:LESSLIMIT 16.0000E-9;LIMIT  
 16.0000E-9;:TRIGGER:A:LOGIC:PATTERN:DELTATIME 16.0000E-9

## TRIGger:A:LOGic:PATtern:DELTatime

Sets or returns the pattern trigger delta time value. The time value is used as part of the pattern trigger condition to determine if the duration of a logic pattern meets the specified time constraints.

**Group** Trigger

**Syntax** TRIGger:A:LOGIC:PATtern:DELTatime <NR3>  
 TRIGger:A:LOGIC:PATtern:DELTatime?

**Arguments** <NR3> is a floating point value with exponent that sets the pattern trigger time value. This argument has a range of 39.6E-9 (39.6 ns) to 10.0E0 (10 s), in increments of 13.2 ns. Values that are not an increment of 13.2 ns are rounded to the nearest correct value.

**Examples** TRIGGER:A:LOGIC:PATTERN:DELTATIME 71.28E-8 sets the pattern trigger delta time value to 712.8 ns.

## TRIGger:A:LOGic:PATtern:INPut:D<x>

Sets or returns the A logic trigger input for the specified digital channel <x>, where x is the channel number. This command species the logic value used when the pattern trigger detects the threshold level.

|                  |  |
|------------------|--|
| <b>Group</b>     | Trigger  |
| <b>Syntax</b>    | TRIGger:A:LOGIC:PATtern:INPut:D<x> {HIGH LOW X}<br>TRIGger:A:LOGIC:PATtern:INPut:D<x>?           |
| <b>Arguments</b> | HIGH specifies a logic high.<br>LOW specifies a logic low.<br>X specifies a “do not care” state. |

## TRIGger:A:LOGIC:PATtern:WHEN

Sets or returns the pattern logic condition on which to trigger the oscilloscope.

|                  |  |
|------------------|--|
| <b>Group</b>     | Trigger  |
| <b>Syntax</b>    | TRIGger:A:LOGIC:PATtern:WHEN<br>{TRUE FALSE LESSThan MOREThan EQUa1 UNEQUa1}<br>TRIGger:A:LOGIC:PATtern:WHEN?  |
| <b>Arguments</b> | TRUE triggers the oscilloscope when the pattern becomes true.<br>FALSE triggers the oscilloscope when the pattern becomes false.<br>LESSTHAN triggers the oscilloscope when the input pattern is true for a time period less than the time period specified in TRIGGER:A:LOGIC:PATTERN:DELTATIME.<br>MORETHAN triggers the oscilloscope when the input pattern is true for a time period more (greater) than the time period specified in TRIGGER:A:LOGIC:PATTERN:DELTATIME.<br>EQUAL triggers the oscilloscope when the input pattern is true for a time period equal to the time period specified in TRIGGER:A:LOGIC:PATTERN:DELTATIME, within a $\pm 5\%$ tolerance.<br>UNEQUAL triggers the oscilloscope when the input pattern is true for a time period greater than or less than (not equal to) the time period specified in TRIGGER:A:LOGIC:PATTERN:DELTATIME, within a $\pm 5\%$ tolerance. |
| <b>Examples</b>  | TRIGGER:A:LOGIC:PATTERN:WHEN LESSTHAN sets the oscilloscope to trigger when the pattern is true for a time period less than the pattern trigger delta time setting.  |



## TRIGger:A:LOGic:PATtern:WHEn:LESSLimit

Sets or returns the maximum time that the selected pattern may be true and still generate an A logic pattern trigger.

|                  |   |
|------------------|---|
| <b>Group</b>     | Trigger   |
| <b>Syntax</b>    | TRIGger:A:LOGic:PATtern:WHEn:LESSLimit <NR3><br>TRIGger:A:LOGic:PATtern:WHEn:LESSLimit?   |
| <b>Arguments</b> | <NR3> specifies the maximum amount of time to hold the pattern true.  |
| <b>Examples</b>  | <p>TRIGGER:A:LOGIC:PATTERN:WHEN:LESSLIMIT 10.0E+00 sets the maximum time that the selected pattern may hold true (and generate an A logic pattern trigger) to 10 s.</p> <p>TRIGGER:A:LOGIC:PATTERN:WHEN:LESSLIMIT? might return<br/>TRIGGER:A:LOGIC:PATTERN:WHEN:LESSLIMIT 8.0000E-9 indicating that the selected pattern may hold true for up to 8 ns and still generate an A logic pattern trigger.</p> |

## TRIGger:A:LOGic:PATtern:WHEn:MORELimit

Sets or returns the minimum time that the selected pattern may be true and still generate an A logic pattern trigger.

|                  |  |
|------------------|--|
| <b>Group</b>     | Trigger  |
| <b>Syntax</b>    | TRIGger:A:LOGic:PATtern:WHEn:MORELimit <NR3><br>TRIGger:A:LOGic:PATtern:WHEn:MORELimit?  |
| <b>Arguments</b> | <NR3> specifies the minimum amount of time to hold the pattern true.   |
| <b>Examples</b>  | <p>TRIGGER:A:LOGIC:PATTERN:WHEN:MORELIMIT 10.0E+00 sets the minimum time that the selected pattern may hold true (and generate an A logic pattern trigger) to 10 s.</p> <p>TRIGGER:A:LOGIC:PATTERN:WHEN:MORELIMIT? might return<br/>TRIGGER:A:LOGIC:PATTERN:WHEN:MORELIMIT 8.0000E-9 indicating that the selected pattern must hold true for at least 8 ns to generate an A logic pattern trigger.</p> |

## TRIGger:A:LOGIc:THReshold:(CH<x>|D<x>)

This command sets or queries the logic trigger threshold voltage for both analog and digital channels (for MSO models), specified by x.

**Group** Trigger

**Syntax** TRIGger:A:LOGIc:THResho]d:(CH<x>|D<x>) {<NR3>|ECL|TTL}  
TRIGger:A:LOGIc:THResho]d:(CH<x>|D<x>)?

**Arguments** <NR3> specifies the threshold voltage, in volts.  
ECL specifies a preset ECL high level of -1.3 V.  
TTL specifies a preset TTL high level of 1.4 V.

**Examples** TRIGGER:A:LOGIC:THRESHOLD:CH2 3.0E-3 sets the A logic trigger threshold voltage for Channel 2 to 3 mV.  
TRIGGER:A:LOGIC:THRESHOLD:CH3? might return  
:TRIGGER:A:LOGIC:THRESHOLD:CH3 1.2000E+00, indicating that the A logic trigger threshold voltage for Channel 3 is 1.2 V.

## TRIGger:A:LOWerthreshold:CH<x>

Sets or returns the lower threshold for the channel selected. Each channel can have an independent level. Used in Runt and Slew Rate triggers as the lower threshold. Used for all other trigger types as the single level/threshold.

**Group** Trigger

**Syntax** TRIGger:A:LOWerthresho]d:CH<x> {ECL|TTL|<NR3>}  
TRIGger:A:LOWerthresho]d:CH<x>?

**Related Commands** [TRIGger:A:LEVel:CH<x>](#)

**Arguments** ECL specifies a preset ECL high level of -1.3 V.  
TTL specifies a preset TTL high level of 1.4 V.  
<NR3> is the clock level, in volts.

- Examples** TRIGGER:A:LOWERTHRESHOLD:CH2 50E-3 sets the lower limit threshold for CH2 of the pulse runt trigger to 50 mV.
- TRIGGER:A:LOWERTHRESHOLD:CH2? might return :TRIGGER:A:LOWERTHRESHOLD:CH2 1.2000E-01 indicating that the lower limit threshold for CH2 of the pulse runt trigger is set to 120 mV.

## TRIGger:A:LOWerthreshold{:EXT|:AUX}

Sets or returns the lower threshold for the Auxiliary Input. It is similar to [TRIGger:A:LEVel:AUXin](#).

- Group** Trigger
- Syntax** TRIGger:A:LOWerthreshold{:EXT|:AUX} {<NR3>|ECL|TTL}  
TRIGger:A:LOWerthreshold{:EXT|:AUX}?
- Arguments** ECL specifies a preset ECL high level of -1.3 V.  
TTL specifies a preset TTL high level of 1.4 V.  
<NR3> specifies the threshold level in volts.

## TRIGger:A:MODE

Sets or returns the A trigger mode.

- Group** Trigger
- Syntax** TRIGger:A:MODE {AUTO|NORMa1}  
TRIGger:A:MODE?
- Related Commands** [TRIGger:A:LEVel](#)
- Arguments** AUTO generates a trigger if one is not detected within a specified time period.  
NORMa1 waits for a valid trigger event.
- Examples** TRIGGER:A:MODE NORMAL specifies that a valid trigger event must occur before a trigger is generated.

TRIGGER:A:MODE ? might return :TRIGGER:A:MODE NORMAL indicating that a valid trigger event must occur before a trigger is generated.

## TRIGger:A:PULse? (Query Only)

Returns the A pulse trigger parameters.

**Group** Trigger

**Syntax** TRIGger:A:PULse?

**Related Commands** [TRIGger:A:EDGE?](#), [TRIGger:A:LOGIc?](#)

**Examples** TRIGGER:A:PULSE? might return :TRIGGER:A:PULSE:CLASS TRANSITION

## TRIGger:A:PULse:CLAss

Sets or returns the type of pulse on which to trigger.

**Group** Trigger

**Syntax** TRIGger:A:PULse:CLAss {RUNT|WIDTH|TRANSITION}  
TRIGger:A:PULse:CLAss?

**Related Commands** [TRIGger:A:RUNT?](#), [TRIGger:A:PULSEWIDTH?](#), [TRIGger:A{:TRANSITION|RISEFall}?](#), [TRIGger:A:TYPE](#)

**Arguments** RUNT triggers when a pulse crosses the first preset voltage threshold but does not cross the second preset threshold before recrossing the first.

WIDTH triggers when a pulse is found that has the specified polarity and is either inside or outside the specified time limits.

TRANSITION triggers when a pulse crosses both thresholds in the same direction as the specified polarity and the transition time between the two threshold crossings is greater or less than the specified time delta.

**Examples** TRIGGER:A:PULSE:CLASS WIDTH specifies a width pulse for the A trigger.  
 TRIGGER:A:PULSE:CLASS? might return :TRIGGER:A:PULSE:CLASS WIDTH indicating that a pulse was found that is of the specified polarity and width.

## TRIGger:A:PULSEWIDTH? (Query Only)

Returns the width parameters for the pulse width trigger.

**Group** Trigger

**Syntax** TRIGger:A:PULSEWIDTH?

**Examples** TRIGGER:A:PULSEWIDTH? might return  
 :TRIGGER:A:PULSEWIDTH:POLARITY POSITIVE;WHEN LESSTHAN;WIDTH  
 8.0000E-9

## TRIGger:A:PULSEWidth:POLarity

Sets or returns the polarity for the width trigger.

**Group** Trigger

**Syntax** TRIGger:A:PULSEwidth:POLarity {NEGative|POSitive}  
 TRIGger:A:PULSEwidth:POLarity?

**Arguments** NEGative specifies a negative pulse.  
 POSitive specifies a positive pulse.

**Examples** TRIGGER:A:PULSEWIDTH:POLARITY NEGATIVE sets the pulse polarity to negative.  
 TRIGGER:A:PULSEWIDTH:POLARITY? might return  
 :TRIGGER:A:WIDTH:POLARITY POSITIVE indicating a positive pulse.

## TRIGger:A:PULSEWidth:SOUrce

Sets or returns the source for the pulse-width trigger.

|                  |   |
|------------------|---|
| <b>Group</b>     | Trigger   |
| <b>Syntax</b>    | TRIGger:A:PULSEwidth:SOURce {{CH1 CH2 CH3 CH4 D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15 LINE EXT}}<br>TRIGger:A:PULSEwidth:SOURce?  |
| <b>Arguments</b> | CH1–CH4 specifies an analog input channel as the A edge trigger source.<br>D0–D15 specifies an digital input channel as the A edge trigger source.<br>EXT specifies an external trigger using the Aux In connector located on the front panel of the oscilloscope.<br>LINE specifies AC line voltage. |
| <b>Examples</b>  | TRIGGER:A:PULSEWIDTH:SOURCE CH1 sets channel 1 as the pulse width source.<br>TRIGGER:A:PULSEWIDTH:SOURCE? might return :TRIGGER:A:PULSEWIDTH:SOURCE CH1 indicating that channel 1 is the pulse width source.  |

## TRIGger:A:PULSEWidth:WHEN

Sets or returns whether to trigger on a pulse that meets, falls outside, or within the specified range of limits.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Trigger   |
| <b>Syntax</b>           | TRIGger:A:PULSEwidth:WHEN {LESSthan MOREthan EQUal UNEQUal}<br>TRIGger:A:PULSEwidth:WHEN?   |
| <b>Related Commands</b> | <a href="#">TRIGger:A:PULSEWidth:WIDTH</a>  |
| <b>Arguments</b>        | LESSthan argument sets the oscilloscope to trigger if a pulse is detected with width less than the time set by the <a href="#">TRIGger:A:PULSEWidth:WIDTH</a> command.<br>MOREthan argument sets the oscilloscope to trigger if a pulse is detected with width more than the time set by the <a href="#">TRIGger:A:PULSEWidth:WIDTH</a> command.<br>EQUal argument sets the oscilloscope to trigger if a pulse is detected with width equal to the time period specified in <a href="#">TRIGger:A:PULSEWidth:WIDTH</a> within a $\pm 5\%$ tolerance.<br>UNEQUal argument sets the oscilloscope to trigger if a pulse is detected with width greater than or less than (but not equal) the time period specified in <a href="#">TRIGger:A:PULSEWidth:WIDTH</a> within a $\pm 5\%$ tolerance. |

**Examples** TRIGGER:A:PULSEWIDTH:WHEN LESSTHAN specifies that the duration of the A pulse will fall within defined high and low limits.

TRIGGER:A:PULSEWIDTH:WHEN? might return :TRIGGER:A:PULSEWIDTH:WHEN THAN indicating that the conditions for generating a width trigger.

## TRIGger:A:PULSEWidth:WIDth

Sets or returns the width setting for the pulse width trigger.

**Group** Trigger

**Syntax** TRIGger:A:PULSEwidth:WIDth <NR3>  
TRIGger:A:PULSEwidth:WIDth?

**Related Commands** [TRIGger:A:PULSEWidth:WHEn](#)

**Arguments** <NR3> specifies the pulse width in seconds.

**Examples** TRIGGER:A:PULSEWIDTH:WIDTH 5.0E-6 sets the pulse width to 5  $\mu$ s.

TRIGGER:A:PULSEWIDTH:WIDTH? might return :TRIGGER:A:PULSEWIDTH:WIDTH 2.0000E-9 indicating that the pulse width is set to 2 ns.

## TRIGger:A:RUNT? (Query Only)

Returns the current A runt trigger parameters.

**Group** Trigger

**Syntax** TRIGger:A:RUNT?

**Examples** TRIGGER:A:RUNT? might return :TRIGGER:A:RUNT:SOURCE CH1;POLARITY POSITIVE;WHEN OCCURS;WIDTH 4.0000E-9.

## TRIGger:A:RUNT:POLarity

Sets or returns the polarity for the runt trigger.

|                  |   |
|------------------|---|
| <b>Group</b>     | Trigger   |
| <b>Syntax</b>    | TRIGger:A:RUNT:POLARity {EITHer NEGative POSitive}<br>TRIGger:A:RUNT:POLARity?  |
| <b>Arguments</b> | <p>POSitive indicates that the rising edge crosses the low threshold and the falling edge recrosses the low threshold without either edge ever crossing the high threshold.</p> <p>NEGative indicates that the falling edge crosses the high threshold and the rising edge recrosses the high threshold without either edge ever crossing the low threshold.</p> <p>EITHer triggers on a runt of either polarity.</p> |
| <b>Examples</b>  | <p>TRIGGER:A:RUNT:POLARITY NEGATIVE specifies that the polarity of the A pulse runt trigger is negative.</p> <p>TRIGGER:A:RUNT:POLARITY? might return :TRIGGER:A:RUNT:POLARITY POSITIVE indicating that the polarity of the A pulse runt trigger is positive.</p>   |

## TRIGger:A:RUNT:SOURce

Sets or returns the source for the A runt trigger.

|                  |   |
|------------------|---|
| <b>Group</b>     | Trigger   |
| <b>Syntax</b>    | TRIGger:A:RUNT:SOURce {CH1 CH2 CH3 CH4}<br>TRIGger:A:RUNT:SOURce?   |
| <b>Arguments</b> | CH1–CH4 specifies the input channel number, depending on the model of the oscilloscope.   |
| <b>Examples</b>  | <p>TRIGGER:A:RUNT:SOURCE CH4 sets channel 4 as the source for the A pulse trigger.</p> <p>TRIGGER:A:RUNT:SOURCE? might return :TRIGGER:A:RUNT:SOURCE CH2 indicating that channel 2 is the source for the A pulse trigger.</p> |

## TRIGger:A:RUNT:WHEn

Sets or returns the type of pulse width the trigger checks for when it detects a runt.



|                         |  |
|-------------------------|--|
| <b>Group</b>            | Trigger  |
| <b>Syntax</b>           | TRIGger:A:RUNT:WHEN {LESSthan MOREthan EQua1 UNEQua1 OCCURS}<br>TRIGger:A:RUNT:WHEN?   |
| <b>Related Commands</b> | <a href="#">TRIGger:A:RUNT:WIDth</a>   |
| <b>Arguments</b>        | <p>OCCURS argument specifies a trigger event if a runt of any detectable width occurs.</p> <p>LESSthan argument sets the oscilloscope to trigger if the a runt pulse is detected with width less than the time set by the <a href="#">TRIGger:A:RUNT:WIDth</a> command.</p> <p>MOREthan argument sets the oscilloscope to trigger if the a runt pulse is detected with width more than the time set by the <a href="#">TRIGger:A:RUNT:WIDth</a> command.</p> <p>EQua1 argument sets the oscilloscope to trigger if a runt pulse is detected with width equal to the time period specified in <a href="#">TRIGger:A:RUNT:WIDth</a> within a <math>\pm 5\%</math> tolerance.</p> <p>UNEQua1 argument sets the oscilloscope to trigger if a runt pulse is detected with width greater than or less than (but not equal to) the time period specified in <a href="#">TRIGger:A:RUNT:WIDth</a> within a <math>\pm 5\%</math> tolerance.</p> |
| <b>Examples</b>         | <p>TRIGGER:A:RUNT:WHEN THAN sets the runt trigger to occur when the oscilloscope detects a runt in a pulse wider than the specified width.</p> <p>TRIGGER:A:RUNT:WHEN? might return :TRIGGER:A:PULSE:RUNT:WHEN OCCURS indicating that a runt trigger will occur if the oscilloscope detects a runt of any detectable width.</p>  |

## TRIGger:A:RUNT:WIDth

Sets or returns the width for a runt trigger.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Trigger   |
| <b>Syntax</b>           | TRIGger:A:RUNT:WIDth <NR3><br>TRIGger:A:RUNT:WIDth? |
| <b>Related Commands</b> | <a href="#">TRIGger:A:RUNT:WHEn</a>                 |
| <b>Arguments</b>        | <NR3> specifies the minimum width, in seconds.      |

- Examples** TRIGGER:A:RUNT:WIDTH 15E-6 sets the minimum width of the pulse runt trigger to 15  $\mu$ s.
- TRIGGER:A:RUNT:WIDTH? might return :TRIGGER:A:PULSE:RUNT:WIDTH 2.0000E-09 indicating that the minimum width of a pulse runt trigger is 2 ns.

## TRIGger:A:SETHold? (Query Only)

Returns the clock edge polarity, voltage threshold and source input; data voltage threshold and source; and both setup and hold times for setup and hold violation triggering.

**Group** Trigger

**Syntax** TRIGger:A:SETHold?

- Examples** TRIGGER:A:SETHOLD? might return  
:TRIGGER:A:SETHOLD:CLOCK:SOURCE CH1;EDGE RISE;THRESHOLD  
100.0000E-3;:TRIGGER:A:SETHOLD:DATA:SOURCE CH2;THRESHOLD  
80.0000E-3;:TRIGGER:A:SETHOLD:HOLDTIME 20.0000E-9;SETTIME  
8.0000E-9

## TRIGger:A:SETHold:CLOCK? (Query Only)

Returns the clock edge polarity, voltage threshold, and source input for setup and hold triggering.

**Group** Trigger

**Syntax** TRIGger:A:SETHold:CLOCK?

- Examples** TRIGGER:A:SETHOLD:CLOCK? might return  
:TRIGGER:A:SETHOLD:CLOCK:SOURCE EXT;EDGE FALL;THRESHOLD  
1.4000

## TRIGger:A:SETHold:CLOCK:EDGE

Sets or returns the clock edge polarity for setup and hold triggering.

|                  |   |
|------------------|---|
| <b>Group</b>     | Trigger   |
| <b>Syntax</b>    | TRIGger:A:SETHold:CLOCK:EDGE {FALL RISe}<br>TRIGger:A:SETHold:CLOCK:EDGE?   |
| <b>Arguments</b> | FALL specifies polarity as the clock falling edge.<br>RISe specifies polarity as the clock rising edge.   |
| <b>Examples</b>  | TRIGGER:A:SETHOLD:CLOCK:EDGE RISE specifies the polarity as the clock rising edge.<br><br>TRIGGER:A:SETHOLD:CLOCK:EDGE? might return :TRIGGER:A:SETHOLD:CLOCK:EDGE RISE indicating that polarity is specified as the clock rising edge. |

## TRIGger:A:SETHold:CLOCK:SOURce

Sets or returns the clock source for the setup and hold triggering.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Trigger   |
| <b>Syntax</b>           | TRIGger:A:SETHold:CLOCK:SOURce {CH1 CH2 CH3 CH4 D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15}<br>TRIGger:A:SETHold:CLOCK:SOURce?   |
| <b>Related Commands</b> | <a href="#">TRIGger:A:SETHold:DATA:SOURce</a>   |
| <b>Arguments</b>        | CH1-CH4 specifies the input channel number.<br>D0-D15 specifies the input digital channel number.   |
| <b>Examples</b>         | TRIGGER:A:SETHOLD:CLOCK:SOURCE CH1 specifies channel 1 as the clock input for setup and hold input.<br><br>TRIGGER:A:SETHOLD:CLOCK:SOURCE? might return :TRIGGER:A:SETHOLD:CLOCK:SOURCE CH4 indicating that channel 4 is the clock source for the setup and hold trigger input. |

## TRIGger:A:SETHold:CLOCK:THReshold

Sets or returns the clock voltage threshold for the setup and hold trigger.

|                  |  |
|------------------|--|
| <b>Group</b>     | Trigger  |
| <b>Syntax</b>    | TRIGGER:A:SETHold:CLOCK:THRESHold {<NR3> TTL ECL}<br>TRIGGER:A:SETHold:CLOCK:THRESHold?  |
| <b>Arguments</b> | TTL specifies a preset TTL high level of 1.4 V.<br>ECL specifies a preset ECL high level of -1.3 V.<br><NR3> is the clock level, in volts.   |
| <b>Examples</b>  | TRIGGER:A:SETHOLD:CLOCK:THRESHOLD TTL specifies the preset TTL value of 1.4 V as the clock threshold for the setup and hold trigger.<br>TRIGGER:A:SETHOLD:CLOCK:THRESHOLD? might return<br>:TRIGGER:A:LOGIC:SETHOLD:CLOCK:THRESHOLD 1.2000E+00<br>indicating that the clock threshold for the setup and hold trigger is 1.2 V. |

## TRIGGER:A:SETHold:DATA? (Query Only)

Returns the voltage threshold and data source for the setup and hold trigger.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Trigger   |
| <b>Syntax</b>           | TRIGGER:A:SETHold:DATA?   |
| <b>Related Commands</b> | <a href="#">TRIGGER:A:SETHold:CLOCK?</a>  |
| <b>Examples</b>         | TRIGGER:A:SETHOLD:DATA? might return<br>:TRIGGER:A:SETHOLD:DATA:SOURCE CH2;THRESHOLD 80.0000E-3 |

## TRIGGER:A:SETHold:DATA:SOURce

Sets or returns the data source for the setup and hold trigger. You cannot specify the same source for both clock and data.

For DPO models, you can specify only a single data source. Data sources for DPO models may be one of CH1-CH4 or D0-D15.

|              |         |
|--------------|---------|
| <b>Group</b> | Trigger |
|--------------|---------|

**Syntax** DPO Models:  
`TRIGger:A:SETHold:DATA:SOURCE {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}`  
`TRIGger:A:SETHold:DATA:SOURCE?`

**Related Commands** [TRIGger:A:SETHold:CLOCK:SOUrce](#)

**Arguments** DPO Models:  
 <wfm> specifies the source channel number and is one of CH1–CH4, and D0–D15. You can specify only one waveform on a DPO.

**Examples** `TRIGGER:A:SETHOLD:DATA:SOURCE CH1` sets channel 1 as the clock source for the setup and hold trigger.

`TRIGGER:A:SETHOLD:DATA:SOURCE?` might return  
`:TRIGGER:A:LOGIC:SETHOLD:DATA:SOURCE CH2` indicating that channel 2 is the current clock source for the setup and hold trigger.

## TRIGger:A:SETHold:DATA:THReshold

Sets or returns the data voltage threshold for setup and hold trigger. The digital channels do not have independent trigger levels. Channels D0–D7 share one common trigger level and D8–D15 share another common trigger level. For example changing the trigger level for D3 changes the levels for D0–D7.

**Group** Trigger

**Syntax** `TRIGger:A:SETHold:DATA:THReshold {<NR3>|TTL|ECL}`  
`TRIGger:A:SETHold:DATA:THReshold?`

**Arguments** TTL specifies the preset TTL high level of 1.4 V.  
 ECL specifies the preset ECL high level of –1.3 V.  
 <NR3> is the setup and hold data level, in V.

**Examples** `TRIGGER:A:SETHOLD:DATA:THRESHOLD TTL` specifies the preset high level of 1.4 V as the current data voltage level for the setup and hold trigger.

`TRIGGER:A:SETHOLD:DATA:THRESHOLD?` might return  
`:TRIGGER:A:SETHOLD:DATA:THRESHOLD 1.2000E+00` indicating that 1.2 V is the current data voltage level for the setup and hold trigger.

## TRIGger:A:SETHold:HOLDTime

Sets or returns the hold time for setup and hold violation triggering.

**Group** Trigger

**Syntax** TRIGger:A:SETHold:HOLDTime <NR3>  
TRIGger:A:SETHold:HOLDTime?

**Arguments** <NR3> specifies the hold time setting in seconds. Positive values for hold time occur after the clock edge. Negative values occur before the clock edge.

**Examples** TRIGGER:A:SETHOLD:HOLDTIME 3.0E-3 sets the hold time for the setup and hold trigger to 3 ms.

TRIGGER:A:SETHOLD:HOLDTIME? might return  
:TRIGGER:A:SETHOLD:HOLDTIME 2.0000E-09 indicating that the current hold time for the setup and hold trigger is 2 ns.

## TRIGger:A:SETHold:SETTime

Sets or returns the setup time for setup and hold violation triggering.

**Group** Trigger

**Syntax** TRIGger:A:SETHold:SETTime <NR3>  
TRIGger:A:SETHold:SETTime?

**Arguments** <NR3> specifies the setup time for setup and hold violation triggering.

**Examples** TRIGGER:A:SETHOLD:SETTIME 3.0E-6 specifies that the current setup time for setup and hold trigger is 3  $\mu$ s.

TRIGGER:A:SETHOLD:SETTIME? might return  
:TRIGGER:A:LOGIC:SETHOLD:SETTIME 2.0000E-09 indicating that the current setup time for setup and hold trigger is 2 ns.

## TRIGger:A:SETHold:THReshold:CH<x>

Sets or queries the threshold for the channel specified by x. Affects all trigger types using the channel.

|                  |  |
|------------------|--|
| <b>Group</b>     | Trigger  |
| <b>Syntax</b>    | TRIGger:A:SETHold:THReshold:CH<x> {<NR3> ECL TTL}<br>TRIGger:A:SETHold:THReshold:CH<x>?  |
| <b>Arguments</b> | <NR3> specifies the threshold voltage, in volts.<br>ECL specifies a preset ECL high level of -1.3 V.<br>TTL specifies a preset TTL high level of 1.4 V.  |
| <b>Examples</b>  | TRIGGER:A:SETHOLD:THRESHOLD:CH1 1.5 sets the channel 1 threshold to 1.5 volts.<br><br>TRIGGER:A:SETHOLD:THRESHOLD:CH1? might return<br>TRIGGER:A:SETHOLD:THRESHOLD:CH1 0.0E+0 indicating that the channel 1 threshold is set to 0.0 volts. |

## TRIGger:A:SETHold:THReshold:D<x>

Sets the Threshold for the Selected Digital Channel. Affects all trigger types using the Digital Channel.

|                  |  |
|------------------|--|
| <b>Group</b>     | MSO  |
| <b>Syntax</b>    | TRIGger:A:SETHold:THReshold:D<x> {<NR3> ECL TTL}<br>TRIGger:A:SETHold:THReshold:D<x>?  |
| <b>Arguments</b> | <NR3> specifies the threshold voltage, in volts.<br>ECL specifies a preset ECL high level of -1.3 V.<br>TTL specifies a preset TTL high level of 1.4 V.  |
| <b>Examples</b>  | TRIGGER:A:SETHOLD:THRESHOLD:D1 ECL sets the threshold to ECL levels.<br><br>TRIGGER:A:SETHOLD:THRESHOLD:D1? might return<br>TRIGGER:A:SETHOLD:THRESHOLD:D1 -1.3000 indicating that the threshold is set to -1.3 volts. |

## TRIGger:A{:TRANSition|:RISEFall}? (Query Only)

Returns transition time trigger parameters.

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Trigger   |
| <b>Syntax</b>           | TRIGger:A{:TRANSition :RISEFall}?   |
| <b>Related Commands</b> | TRIGger:A:UPPerthreshold:CH<x> , TRIGger:A:LOWerthreshold:CH<x>   |
| <b>Examples</b>         | TRIGGER:A::TRANSITION? might return<br>:TRIGGER:A:TRANSITION:POLARITY POSITIVE;WHEN<br>SLOWER;DELTATIME 8.0000E-9 |

## TRIGger:A{:TRANSition|:RISEFall}:DELTatime

Sets or returns the delta time used in calculating the transition value for the transition trigger.

|                  |   |
|------------------|---|
| <b>Group</b>     | Trigger   |
| <b>Syntax</b>    | TRIGger:A{:TRANSition :RISEFall}:DELTatime <NR3><br>TRIGger:A{:TRANSition :RISEFall}:DELTatime?   |
| <b>Arguments</b> | <NR3> specifies the delta time, in seconds.   |
| <b>Examples</b>  | TRIGGER:A:TRANSITION:DELTATIME 15E-6 sets the delta time of the transition trigger to 15 $\mu$ s.<br><br>TRIGGER:A:TRANSITION:DELTATIME? might return :TRIGGER:A:TRANSITION:DELTATIME 2.0000E-09 indicating that the delta time of the transition trigger is set to 2 ns. |

## TRIGger:A{:TRANSition|:RISEFall}:POLarity

Sets or returns the polarity for the transition trigger.

|               |   |
|---------------|---|
| <b>Group</b>  | Trigger   |
| <b>Syntax</b> | TRIGger:A{:TRANSition :RISEFall}:POLarity<br>{EITher NEGative POSitive}<br>TRIGger:A{:TRANSition :RISEFall}:POLarity? |



- Arguments** POSitive indicates that a pulse edge must traverse from the lower (most negative) to higher (most positive) level for transition triggering to occur.
- NEGative indicates that a pulse edge must traverse from the upper (most positive) to lower (most negative) level for transition triggering to occur.
- EITHer indicates either positive or negative polarity.
- Examples** TRIGGER:A:TRANSITION:POLARITY NEGATIVE sets the transition polarity to negative.
- TRIGGER:A:TRANSITION:POLARITY? might return :TRIGGER:A:TRANSITION:POLARITY EITHER indicating that the polarity can be either positive or negative.

## TRIGger:A{:TRANSition|:RISEFall}:SOURce

Sets or returns the source for transition trigger.

- Group** Trigger
- Syntax** TRIGger:A{:TRANSition|:RISEFall}:SOURce {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}  
TRIGger:A{:TRANSition|:RISEFall}:SOURce?
- Arguments** CH1–CH4 specifies one of the input channels.
- D0–D15 specifies one of the input digital channels.
- Examples** TRIGGER:A:TRANSITION:SOURCE CH4 sets channel 4 as the source for the transition trigger.
- TRIGGER:A:TRANSITION:SOURCE? might return :TRIGGER:A:TRANSITION:SOURCE CH2 indicating that channel 2 is the source for the A transition trigger.

## TRIGger:A{:TRANSition|:RISEFall}:WHEn

Sets or returns whether to check for a transitioning signal that is faster or slower than the specified delta time.

- Group** Trigger

**Syntax** TRIGger:A{:TRANSition|:RISEFall}:WHEN  
 {SLOWer|FASTER|EQua1|UNEQua1}  
 TRIGger:A{:TRANSition|:RISEFall}:WHEN?

**Arguments** FASTER sets the trigger to occur when the signal transition time is faster than the time set by TRIGger:A{TRANSition|:RISEFall}:DELTatime.  
 SLOWer sets the trigger to occur when the signal transition time is slower than the time set by.TRIGger:A{TRANSition|:RISEFall}:DELTatime.  
 EQua1 sets the trigger to occur when the signal transition time is equal to the time set by TRIGger:A{TRANSition|:RISEFall}:DELTatime.  
 UNEQua1 sets the trigger to occur when the signal transition time is not equal to the time set by TRIGger:A{TRANSition|:RISEFall}:DELTatime.

**Examples** TRIGGER:A:TRANSITION:WHEN SLOWER sets the trigger to occur when the signal transition time is slower than the time set by.TRIGger:A{TRANSition|:RISEFall}:DELTatime.  
 TRIGGER:A:TRANSITION:WHEN? might return :TRIGGER:A:TRANSITION:WHEN FASTER

## TRIGger:A:TYPE

Sets or returns the type of A trigger. The five types of triggers are of Edge, Logic, Pulse, Serial, and Video. Logic and Pulse triggers contain classes. Logic triggers consist of State, Pattern, and SetHold classes; Pulse triggers consist of Runt, Width, and Transition logic classes. Once you have set the trigger type, you may also need to identify the associated trigger class. For details on selecting Logic and Pulse trigger classes, see [TRIGger:A:LOGic:CLAss](#) and [TRIGger:A:PULse:CLAss](#) respectively.

**Group** Trigger

**Syntax** TRIGger:A:TYPE {EDGE|LOGic|PULSe|BUS|VIDeo}  
 TRIGger:A:TYPE?

**Related Commands** [TRIGger:A:EDGE?](#), [TRIGger:A:LOGic:CLAss](#), [TRIGger:A:PULse:CLAss](#)

**Arguments** EDGE is a normal trigger. A trigger event occurs when a signal passes through a specified voltage level in a specified direction and is controlled by the [TRIGger:A:EDGE?](#) commands.

LOGic specifies that a trigger occurs when specified conditions are met and is controlled by the `TRIGger:A:LOGic?` commands.

PULSe specifies that a trigger occurs when a specified pulse is found and is controlled by the `TRIGger:A:PULse?` commands.

BUS specifies that a trigger occurs when a communications signal is found. Supports CAN, I<sup>2</sup>C, SPI, and RS232 communications signals.

VIDeo specifies that the trigger occurs when a video signal is found.

**Examples** `TRIGGER:A:TYPE EDGE` sets the A trigger type to EDGE.  
`TRIGGER:A:TYPE?` might return `:TRIGGER:A:TYPE PULSE` indicating that the A trigger type is a pulse trigger.

## TRIGger:A:UPPerthreshold:CH<x>

Sets the upper threshold for channel <x>, where x is the channel number. Each channel can have an independent level. Used only for runt and slew rate trigger types.

**Group** Trigger

**Syntax** `TRIGger:A:UPPerthreshold:CH<x> {<NR3>|ECL|TTL}`  
`TRIGger:A:UPPerthreshold:CH<x>?`

**Arguments** <NR3> is the threshold level in volts.  
 ECL specifies a preset ECL high level of -1.3 V.  
 TTL specifies a preset TTL high level of 1.4 V.

**Examples** `TRIGGER:A:UPPERTHRESHOLD:CH2 50E-3` sets the upper limit of the pulse runt trigger to 50 mV for channel 2.  
`TRIGGER:A:UPPERTHRESHOLD:CH2?` might return `:TRIGGER:A:UPPERTHRESHOLD:CH2 1.2000E-01` indicating that the upper limit of the pulse runt trigger is set to 120 mV.

## TRIGger:A:VIDeo? (Query Only)

Returns the A trigger video parameters.

**Group** Trigger

**Syntax** TRIGger:A:VIDeo?

**Examples** TRIGGER:A:VIDEO? might return :TRIGGER:A:VIDEO:POLARITY POSITIVE;SOURCE CH1;STANDARD NTSC;SYNC ALLLINES;HOLDOFF:FIELD 0.0000;:TRIGGER:A:VIDEO:CUSTOM:FORMAT PROGRESSIVE;SCANRATE15K;:TRIGGER:A:VIDEO:LINE 1;HDTV:FORMAT HD1080I60

## TRIGger:A:VIDeo:HOLDoff:FIELD

Sets or returns the video trigger holdoff in terms of video fields.

**Conditions** This command requires a DPO2VID application module.

**Group** Trigger

**Syntax** TRIGger:A:VIDeo:HOLDoff:FIELD <NR3>  
TRIGger:A:VIDeo:HOLDoff:FIELD?

**Arguments** <NR3> argument is a real number from 0.0 to 8.5 in increments of 0.5. The argument sets the number of fields that the oscilloscope waits before rearming the video trigger.

**Examples** TRIGGER:A:VIDEO:HOLDOFF:FIELD? might return :TRIGGER:A:VIDEO:HOLDOFF:FIELD 5 indicating that the oscilloscope is set to wait 5 video fields before rearming the trigger.

TRIGGER:A:VIDEO:HOLDOFF:FIELD 4.5 sets the oscilloscope to wait 4.5 video fields before rearming the trigger.

## TRIGger:A:VIDeo:LINE

Sets or returns the video line number on which the oscilloscope triggers. Use the [TRIGger:A:VIDeo{:SYNC\[:FIELD\]}](#) command to actually trigger the oscilloscope on the line that you specify with this command.

**Conditions** This command requires a DPO2VID application module.

**Group** Trigger

**Syntax** TRIGger:A:VIDeo:LINE <NR1>  
TRIGger:A:VIDeo:LINE?

**Related Commands** TRIGger:A:VIDeo{:SYNC|:FIELD}

**Arguments** <NR1> argument is an integer that sets the video line number on which the oscilloscope triggers. The following table lists the valid choices, depending on the active video standard.

**Table 2-41: Video Line Numbering Ranges**

| Video Standard | Line Number Range |
|----------------|-------------------|
| 525/NTSC       | 1–525             |
| 625/PAL, SECAM | 1–625             |
| SECAM          | 1–625             |

**Examples** TRIGGER:A:VIDEO:LINE 23 sets the oscilloscope to trigger on the line 23.  
TRIGGER:A:VIDEO:LINE ? might return :TRIGGER:A:VIDEO:LINE 10 indicating that the oscilloscope is set to trigger on line 10.

## TRIGger:A:VIDeo:POLarity

Sets or returns the polarity of the A video trigger.

**Conditions** This command requires a DPO2VID application module.

**Group** Trigger

**Syntax** TRIGger:A:VIDeo:POLarity {NEGative|POSitive|NORMal|INVERTed}  
TRIGger:A:VIDeo:POLarity?

**Arguments** POSitive argument sets the oscilloscope to trigger on a positive video sync pulse.  
NEGative argument sets the oscilloscope to trigger on a negative video sync pulse.

- Examples** TRIGGER:A:VIDEO:POLARITY NEGATIVE sets the oscilloscope to trigger on a negative video pulse.
- TRIGGER:A:VIDEO:POLARITY? might return :TRIGGER:A:VIDEO:POLARITY POSITIVE indicating that the oscilloscope is set to trigger on a positive video sync pulse.

## TRIGger:A:VIDeo:SOURce

Sets or returns the source for the A video trigger.

- Group** Trigger
- Syntax** TRIGger:A:VIDeo:SOURce {CH1|CH2|CH3|CH4}  
TRIGger:A:VIDeo:SOURce?
- Arguments** CH1–CH4 specifies the input channel to use as the A video trigger.  
D0–D15 specifies the input channel to use as the A video trigger.
- Examples** TRIGGER:A:VIDEO:SOURCE CH1 sets the source for A video trigger to Channel 1.  
TRIGGER:A:VIDEO:SOURCE? might return :TRIGGER:A:VIDEO:SOURCE CH2 indicating that the source for the A video trigger is set to Channel 2.

## TRIGger:A:VIDeo:STANdard

Sets or returns the standard for the video trigger.

- Group** Trigger
- Syntax** TRIGger:A:VIDeo:STANdard {NTSC|PAL|SECAM}  
TRIGger:A:VIDeo:STANdard?
- Arguments** NTSC sets the oscilloscope to trigger on video signals that meet the NTSC 525/60/2:1 standard (a line rate of 525 lines per frame and a field rate of 60 Hz).
- PAL sets the oscilloscope to trigger on video signals that meet the NTSC 625/50/2:1 standard (a line rate of 625 lines per frame and a field rate of 50 Hz).
- SECAM sets the oscilloscope to trigger on video signals that meet the SECAM standard.

- Examples** TRIGGER:A:VIDEO:STANDARD NTSC sets the oscilloscope to trigger on NTSC-standard video signals.
- TRIGGER:A:VIDEO:STANDARD? might return :TRIGGER:A:VIDEO:STANDARD HDTV indicating that the oscilloscope is set to trigger on an HDTV format.

## TRIGger:A:VIDeo{:SYNC|:FIELD}

Sets or returns the video field or line that the trigger detects.

**Group** Trigger

**Syntax** TRIGger:A:VIDeo{:SYNC|:FIELD}  
{ODD|EVEN|ALLFields|ALLLines|NUMERIC}  
TRIGger:A:VIDeo{:SYNC|:FIELD}?

- Arguments** ODD argument sets the oscilloscope to trigger on interlaced video odd fields.
- EVEN argument sets the oscilloscope to trigger on interlaced video even fields.
- ALLFields argument sets the oscilloscope to trigger on all fields.
- ALLLines argument sets the oscilloscope to trigger on all video lines.
- NUMERIC argument sets the oscilloscope to trigger on the video signal line specified by the [TRIGger:A:VIDeo:LINE](#) command.

- Examples** TRIGGER:A:VIDEO:FIELD EVEN sets the A video trigger so that it will trigger on even fields.
- TRIGGER:A:VIDEO:FIELD? might return :TRIGGER:A:VIDEO:FIELD ALLFIELDS indicating that the A video will trigger on all video fields.

## TRIGger:EXTernal? (Query Only)

Returns all external trigger parameters.

**Group** Trigger

**Syntax** TRIGger:EXTernal?

## TRIGger:EXternal:PRObe

Sets or returns the attenuation factor value of the external probe connector.

**Group** Trigger

**Syntax** TRIGger:EXternal:PRObe <NR3>  
TRIGger:EXternal:PRObe?

**Arguments** <NR3> is the attenuation factor of the probe.

**Examples** TRIGGER:EXTERNAL:PROBE? might return :TRIGGER:EXTERNAL:PROBE  
1.0E1 for a 10X probe.

## TRIGger:EXternal:YUNIts? (Query Only)

Returns the external trigger vertical (Y) units value.

**Group** Trigger

**Syntax** TRIGger:EXternal:YUNIts?

**Examples** TRIGGER:EXTERNAL:YUNITS? might return TRIGGER:EXTERNAL:YUNITS  
"V" if the vertical unit is volts.

## TRIGger:STATE? (Query Only)

Returns the current state of the triggering system.

**Group** Trigger

**Syntax** TRIGger:STATE?

**Related Commands** [TRIGger:A:MODE](#)

**Returns** ARMED indicates that the oscilloscope is acquiring pretrigger information.



AUTO indicates that the oscilloscope is in the automatic mode and acquires data even in the absence of a trigger.

READY indicates that all pretrigger information has been acquired and that the oscilloscope is ready to accept a trigger.

SAVE indicates that the oscilloscope is in save mode and is not acquiring data.

TRIGGER indicates that the oscilloscope triggered and is acquiring the post trigger information.

**Examples** TRIGGER:STATE? might return :TRIGGER:STATE ARMED indicating that the pretrigger data is being acquired.

## \*TST? (Query Only)

Tests (self-test) the interface and returns a 0.

**Group** Miscellaneous

**Syntax** \*TST?

**Examples** \*TST? always returns 0.

## UNLock (No Query Form)

Unlocks the front panel. The command is equivalent to LOCK NONE.

**Group** Miscellaneous

**Syntax** UNLOCK ALL

**Related Commands** [LOCK](#)

**Arguments** ALL specifies that all front-panel buttons and knobs are unlocked.

**Examples** UNLOCK ALL unlocks all front-panel buttons and knobs.

## USBTMC

Returns the USBTMC information used by the USB hosts to determine the instrument interfaces.

**Group** Misc

**Syntax** USBTMC?

## USBDevice:CONFigure

Enables or disables the rear USB port for use with Pictbridge printers.

**Group** PictBridge

**Syntax** USBDevice:CONFigure {DISabled|IMAge|USBTmc}  
USBDevice:CONFigure?

**Arguments** DISabled disables the rear USB port.  
IMAge enables the rear USB port as an SIC device.  
USBTmc enables the rear USB port as a USBTMC device.

**Examples** USBDEVICE:CONFIGURE? might return USBDEVICE:CONFIGURE USBTMC indicating a USBTMC device.  
USBDEVICE:CONFIGURE IMAGE enables the rear USB port as an SIC device

## USBTMC:PRODUCTID:DECimal? (Query Only)

Returns the product ID of the USBTMC device in decimal format.

**Group** Misc

**Syntax** USBTMC:PRODUCTID:DECimal?

**Examples** USBTMC:PRODUCTID:DECIMAL? might return USBTMC:PRODUCTID:DECIMAL 882 indicating that the product ID is 882 in decimal format.

## USBTMC:PRODUCTID:HEXadecimal? (Query Only)

Returns the product ID of the USBTMC device in hexadecimal format.

**Group** Misc

**Syntax** USBTMC:PRODUCTID:HEXadecimal?

**Examples** USBTMC:PRODUCTID:HEXADECIMAL? might return  
USBTMC:PRODUCTID:HEXADECIMAL 0x0372 indicating that the product ID in  
hexadecimal format is 0x0372.

## USBTMC:SERIALnumber? (Query Only)

Returns the serial number of the USBTMC device.

**Group** Misc

**Syntax** USBTMC:SERIALnumber?

**Examples** USBTMC:SERIALNUMBER? might return USBTMC:SERIALNUMBER NO\_SERIAL  
indicating that there is no serial number for the instrument.

## USBTMC:VENDORID:DECimal? (Query Only)

Returns the vendor ID of the USBTMC device in decimal format.

**Group** Misc

**Syntax** USBTMC:VENDORID:DECimal?

**Examples** USBTMC:VENDORID:DECIMAL? might return USBTMC:VENDORID:DECIMAL  
1689 indicating that 1689 is the Vendor ID in decimal format.

## USBTMC:VENDORID:HEXadecimal? (Query Only)

Returns the vendor ID of the USBTMC device in hexadecimal format.

**Group** Misc

**Syntax** USBTMC:VENDORID:HEXadecimal?

**Examples** USBTMC:VENDORID:HEXADECIMAL? might return  
 USBTMC:VENDORID:HEXADECIMAL 0x0699 indicating that 0x0689 is the  
 Vendor ID in hexadecimal format.

## VERBose

Sets or returns the Verbose state that controls the length of keywords on query responses. Keywords can be both headers and arguments.

---

**NOTE.** *This command does not affect IEEE Std 488.2-1987 Common Commands (those starting with an asterisk). This command does affect the verbose state of both the USBTMC and VXI-11 interfaces. Refer to the Introduction for information.*

---

**Group** Miscellaneous

**Syntax** VERBose {OFF|ON|<NR1>}

**Related Commands** [HEADer](#), [\\*LRN?](#), [SET?](#)

**Arguments** OFF sets the Verbose state to false, which returns minimum-length keywords for applicable setting queries.

ON sets the Verbose state to true, which returns full-length keywords for applicable setting queries.

<NR1> a 0 returns minimum-length keywords for applicable setting queries; any other value returns full-length keywords.

**Examples** VERBOSE ON enables the Verbose state.

VERBOSE ? might return :VERB 0 indicating that the Verbose state is disabled.

## \*WAI (No Query Form)

Prevents the oscilloscope from executing further commands or queries until all pending commands that generate an OPC message are complete. This command allows you to synchronize the operation of the oscilloscope with your application program.

**Group** Status and Error

**Syntax** \*WAI

**Related Commands** [BUSY?](#), [\\*OPC](#)

**Examples** \*WAI prevents the oscilloscope from executing any further commands or queries until all pending commands that generate an OPC message are complete.

## WAVFrm? (Query Only)

Returns WFMOutpre? and CURVe? data for the waveform as specified by the DATA:SOURce command. This command is equivalent to sending both WFMOutpre? and CURVe?, with the additional provision that the response to WAVFrm? is guaranteed to provide a synchronized preamble and curve. The source waveform, as specified by :DATA:SOURCE, must be active or the query will not return any data and will generate an error indicator.

---

**NOTE.** For MSO models, reduced resolution waveform data is not available for the digital channels.

---

**Group** Waveform Transfer

**Syntax** WAVFrm?

**Related Commands** [CURVe](#), [DATA:SOURce](#), [WFMOutpre?](#)

**Examples** WAVFRM? might return the waveform data as: :WFMOUTPRE:BYT\_NR 1;BIT\_NR 8;ENCDG ASCII;BN\_FMT RI;BYT\_OR MSB;WFID "Ch1, DC coupling, 100.0mV/div, 4.000us/div, 10000 points, Sample mode";NR\_PT 20;PT\_FMT Y;XUNIT "s";XINCR 4.0000E-9;XZERO -20.0000E-6;PT\_OFF 0;YUNIT

```
"V";YMULT 4.0000E-3;YOFF 0.0000;YZERO 0.0000;:CURVE
2,1,4,2,4,3,0,3,3,3,3,3,3,4,3,5,6,6,7,3
```

## WFMInpre? (Query Only)

Returns the waveform formatting and scaling specifications to be applied to the next incoming CURVe command data.

**Group** Waveform Transfer

**Syntax** WFMInpre?

**Related Commands** [WFMOutpre?](#)

**Examples** WFMINPRE? might return the waveform formatting as :WFMINPRE:BIT\_NR 8;BN\_FMT RI;BYT\_NR 1; BYT\_OR MSB;ENCDG BIN;NR\_PT 500;PT\_FMT Y; PT\_OFF 0;XINCR 2.0000E-6;XZERO 1.7536E-6; XUNIT "s";YMULT 1.0000E-3;YOFF 0.0000; YZERO 0.0000;YUNIT "v"

## WFMInpre:BIT\_Nr

Sets or returns the number of bits per binary waveform point for the incoming waveform. Changing the value of [WFMInpre:BIT\\_Nr](#) also changes the value of [WFMInpre:BYT\\_Nr](#).

**Group** Waveform Transfer

**Syntax** WFMInpre:BIT\_Nr <NR1>  
WFMInpre:BIT\_Nr?

**Related Commands** [WFMInpre:BYT\\_Nr](#)

**Arguments** <NR1> number of bits per data point can be 8 or 16.

**Examples** WFMINPRE:BIT\_NR 16 sets the number of bits per waveform point to 16, for incoming data.

WFMINPRE:BIT\_NR? might return :WFMINPRE:BIT\_NR 8 indicating that incoming waveform data uses 8 bits per waveform point.

## WFMinpre:BN\_Fmt

Sets or returns the format of binary data for incoming waveforms.

**Group** Waveform Transfer

**Syntax** WFMinpre:BN\_Fmt {RI|RP}  
WFMinpre:BN\_Fmt?

**Related Commands** [WFMinpre:BN\\_Fmt](#)

**Arguments** RI specifies signed integer data point representation.  
RP specifies positive integer data point representation.

**Examples** WFMINPRE:BN\_FMT RP specifies positive integer data point representation.  
WFMINPRE:BN\_FMT? might return :WFMINPRE:BN\_FMT RI indicating that the incoming data is currently interpreted as signed integers.

## WFMinpre:BYT\_Nr

Sets or returns the data width for the incoming waveform. Changing the value of [WFMinpre:BYT\\_Nr](#) also changes the value of [WFMinpre:BIT\\_Nr](#).

**Group** Waveform Transfer

**Syntax** WFMinpre:BYT\_Nr <NR1>  
WFMinpre:BYT\_Nr?

**Related Commands** [WFMinpre:BIT\\_Nr](#)

**Arguments** <NR1> is the number of bytes per data point and can be 1 or 2.

**Examples** WFMINPRE:BYT\_NR 1 sets the number of bytes per incoming waveform data point to 1, which is the default setting.  
WFMINPRE:BYT\_NR? might return :WFMINPRE:BYT\_NR 2 indicating that there are 2 bytes per incoming waveform data point.

## WFMInpre:BYT\_Or

Sets or returns which byte of binary waveform data is expected first for incoming waveform data when data points require than one byte. This specification only has meaning when [WFMInpre:ENCdg](#) is set to BIN and [WFMInpre:BYT\\_Nr](#) is 2.

**Group** Waveform Transfer

**Syntax** WFMInpre:BYT\_Or {LSB|MSB}  
WFMInpre:BYT\_Or?

**Related Commands** [WFMInpre:ENCdg](#), [WFMInpre:BYT\\_Nr](#), [WFMInpre:FILTERFreq](#)

**Arguments** LSB specifies that the least significant byte will be expected first.  
MSB specifies that the most significant byte will be expected first.

**Examples** WFMInPRE:BYT\_OR MSB sets the most significant incoming byte of incoming waveform data to be expected first.  
WFMInPRE:BYT\_OR? might return :WFMInPRE:BYT\_OR LSB indicating that the least significant incoming CURVe data byte will be expected first.

## WFMInpre:COMPosition

Sets and queries the type of waveform data to be transferred to the instrument with the [CURVe](#) command.

**Group** Waveform Transfer

**Syntax** WFMInpre:COMPosition  
{COMPOSITE\_YT|COMPOSITE\_ENV|SINGULAR\_YT}  
WFMInpre:COMPosition?

**Related Commands** [DATA:COMPosition](#)

**Arguments** COMPOSITE\_YT uses the sample part of the composite waveform.  
COMPOSITE\_ENV uses the peak-detect part of the composite waveform.  
SINGULAR\_YT uses the sample part of the singular waveform.



**Examples** `WFMINPRE:COMPOSITION?` might return `:WFMINPRE:COMPOSITION SINGULAR_YT` indicating that the sample part of singular waveform is transferred to the instrument.

`WFMINPRE:COMPOSITIONCOMPOSITE_YT` sets the sample part of composite waveform to be transferred to the instrument.

## WFMinpre:FILTERFreq

Specifies or returns the FilterVu low pass filter frequency, which was applied to the waveform being sent to the instrument.

**Group** Waveform Transfer

**Syntax** `WFMinpre:FILTERFreq {NR1}`  
`WFMinpre:FILTERFreq?`

**Arguments** <NR1> is an signed integer.

**Examples** `WFMINPRE:FILTERFREQ?` might return `:WFMINPRE:FILTERFREQ 200000000` indicating that the FilterVu frequency for the incoming waveform is 200 MHz.

`WFMINPRE:FILTERFREQ 300000000` sets the FilterVu frequency for the incoming waveform to 300 MHz.

## WFMinpre:ENCdg

Sets or returns the type of encoding for incoming waveform data.

**Group** Waveform Transfer

**Syntax** `WFMinpre:ENCdg {ASCIi|BINary}`  
`WFMinpre:ENCdg?`

**Related Commands** [WFMinpre:ENCdg](#)

**Arguments** `ASCIi` specifies that the incoming data is in ASCII format.

`BINary` specifies that the incoming data is in a binary format whose further interpretation requires knowledge of `BYT_NR`, `BIT_NR`, `BN_FMT`, and `BYT_OR`.

- Examples** `WFMINPRE:ENCDG ASC` sets the format of incoming waveform data to ASCII format.
- `WFMINPRE:ENCDG ?` might return `:WFMINPRE:ENCDG BIN` indicating that the incoming waveform data is in binary format.

## WFMinpre:NR\_Pt

Sets or returns the number of data points that are in the incoming waveform record.

**Group** Waveform Transfer

**Syntax** `WFMinpre:NR_Pt <NR1>`  
`WFMinpre:NR_Pt?`

**Related Commands** [CURVe](#), [DATA](#), [DATA:START](#), [DATA:STOP](#), [SAVe:WAVEform](#), [SAVe:WAVEform:FILEFormat](#), [WFMinpre:NR\\_Pt?](#)

**Arguments** `<NR1>` is the number of data points if `WFMinpre:PT_Fmt` is set to `Y`. It is the number of min-max pairs if `WFMinpre:PT_Fmt` is set to `ENV`.

- Examples** `WFMINPRE:NR_PT 10000` specifies that 10000 data points will be expected.
- `WFMINPRE:NR_PT ?` might return `:WFMINPRE:NR_PT 10000` indicating that there are 10000 data points in the expected incoming waveform record.

## WFMinpre:PT\_Fmt

Sets or returns the point format of the incoming waveform data. Regardless of the argument used, the scale, offset, and so on are interpreted similarly. When `ENV` is used, waveform data is interpreted over the min-max pair; when `Y` is used, it is interpreted over a single point.

**Group** Waveform Transfer

**Syntax** `WFMinpre:PT_Fmt {ENV|Y}`  
`WFMinpre:PT_Fmt?`

**Related Commands** [WFMinpre:PT\\_Fmt?](#)

**Arguments** ENV specifies that the waveform is transmitted in envelope mode as maximum and minimum point pairs. Only Y values are explicitly transmitted. Absolute coordinates are given by:

$$X_n = XZ_{Ero} + XIN_{Cr} (n - PT_{_Off})$$

$$Y_{nmax} = YZ_{Ero} + YMULt (yn_{max} - YOFf)$$

$$Y_{nmin} = YZ_{Ero} + YMULt (yn_{min} - YOFf)$$

Y specifies a normal waveform where one ASCII or binary data point is transmitted for each point in the waveform record. Only Y values are explicitly transmitted. Absolute coordinates are given by:

$$X_n = XZ_{Ero} + XIN_{Cr} (n - PT_{_Off})$$

$$Y_n = YZ_{Ero} + YMULt (yn - YOFf)$$

**Examples** WFMINPRE:PT\_FMT ENV sets the incoming waveform data point format to enveloped.

WFMINPRE:PT\_FMT? might return :WFMINPRE:PT\_FMT ENV indicating that the waveform is transmitted as maximum and minimum point pairs.

## WFMInpre:PT\_Off

The set form of this command is ignored. The query form always returns a 0. This command is listed for compatibility with other Tektronix oscilloscopes.

**Group** Waveform Transfer

**Syntax** WFMInpre:PT\_Off <NR1>  
WFMInpre:PT\_Off?

**Arguments** Arguments are ignored.

## WFMInpre:XINcr

Sets or returns the horizontal interval between incoming waveform points in units specified by WFMInpre:XUNit.

**Group** Waveform Transfer

**Syntax**      `WFMinpre:XINcr <NR3>`  
`WFMinpre:XINcr?`

**Related Commands**    [WFMinpre:XUNit](#), [WFMOuppre:XINcr?](#)

**Arguments**          <NR3> is the horizontal interval representation.

**Examples**            `WFMINPRE:XINCR 3E-3` sets the interval between incoming waveform points to 3 ms.

`WFMINPRE:XINCR ?` might return `:WFMINPRE:XINCR 1.0000E-3` indicating that if `WFMinpre:XUNit` is set to "s", there is a 1 ms interval between incoming waveform points.

## WFMinpre:XUNit

Sets or returns the horizontal units of the incoming waveform.

Supported units are:

%, /Hz, A, A/A, A/V, A/W, A/dB, A/s, AA, AW, AdB, As, B, Hz, IRE, S/s, V, V/A, V/V, V/W, V/dB, V/s, VV, VW, VdB, Volts, Vs, W, W/A, W/V, W/W, W/dB, W/s, WA, WV, WW, WdB, Ws, dB, dB/A, dB/V, dB/W, dB/dB, dBA, dBV, dBW, dBdB, day, degrees, div, hr, min, ohms, percent, s

**Group**                Waveform Transfer

**Syntax**              `WFMinpre:XUNit <QString>`  
`WFMinpre:XUNit?`

**Related Commands**    [WFMOuppre:XUNit?](#)

**Arguments**          <QString> contains a maximum of three alpha characters that represent the horizontal unit of measure for the incoming waveform.

**Examples**            `WFMINPRE:XUNIT "HZ"` specifies that the horizontal units for the incoming waveform are hertz.

`WFMINPRE:XUNIT?` might return `:WFMINPRE:XUNIT "s"` indicating that the horizontal units for the incoming waveform are seconds.

## WFMinpre:XZEro

Sets or returns the position value, in XUNits, of the first sample of the incoming waveform.

**Group** Waveform Transfer

**Syntax** WFMinpre:XZEro <NR3>  
WFMinpre:XZEro?

**Related Commands** [WFMinpre:XINcr](#), [WFMinpre:XUNit](#), [WFMOupre:XZEro?](#)

**Arguments** <NR3> argument is the floating point value of the position, in XUNits, of the first sample in the incoming waveform. If XUNits is “s”, <NR3> is the time of the first sample in the incoming waveform.

**Examples** WFMINPRE:XZERO 5.7E-6, which indicates the trigger occurred 5.7 μs before the first sample in the waveform.

WFMINPRE:XZERO? might return :WFMINPRE:XZEro -7.5000E-6 indicating that the trigger occurs 7.5 μs after the first sample in the waveform.

## WFMinpre:YMUlt

Sets or returns the vertical scale factor of the incoming waveform, expressed in YUNits per waveform data point level. For one byte waveform data, there are 256 data point levels. For two byte waveform data there are 65,536 data point levels.

YMUlt, YOFF, and YZEro are used to convert waveform record values to YUNit values using the following formula (where dl is the data level; curve\_in\_dl is a data point in CURVe):

$$\text{value\_in\_units} = ((\text{curve\_in\_dl} - \text{YOFF\_in\_dl}) * \text{YMUlt}) + \text{YZEro\_in\_units}$$

---

**NOTE.** For a given waveform record, YMUlt, YOFF, and YZEro have to be a consistent set, otherwise vertical cursor readouts and vertical measurements may give incorrect results.

---

**Group** Waveform Transfer

**Syntax** WFMinpre:YMUlt <NR3>  
WFMinpre:YMUlt?

|                         |   |
|-------------------------|---|
| <b>Related Commands</b> | <a href="#">DATA:DESTination</a> , <a href="#">WFMinpre:BYT_Nr</a> , <a href="#">WFMinpre:YUNit</a>   |
| <b>Arguments</b>        | <NR3> is the vertical scale factor per digitizing level of the incoming waveform points.  |
| <b>Examples</b>         | <p>WFMINPRE:YMULT? might return :WFMINPRE:YMULT 40.0000E-3, which (if YUNit is "V") indicates that the vertical scale is 40 mV/digitizing level (1V/div for 8-bit data).</p> <p>WFMINPRE:YMULT 20E-3 specifies that (if WFMinpre:YUNit is "V" and WFMinpre:BYT_Nr is 1), the vertical scale is 20 mV/digitizing level (500 mV/div).</p> |

## WFMinpre:YOff

Sets or returns the vertical position of the incoming waveform in digitizing levels. Variations in this number are analogous to changing the vertical position of the waveform.

YMUlt, YOOff, and YZEro are used to convert waveform record values to YUNit values using the following formula (where dl is the data level; curve\_in\_dl is a data point in CURVe):

$$\text{value\_in\_units} = ((\text{curve\_in\_dl} - \text{YOOff\_in\_dl}) * \text{YMUlt}) + \text{YZEro\_in\_units}$$

---

**NOTE.** For a given waveform record, YMUlt, YOOff, and YZEro have to be a consistent set, otherwise vertical cursor readouts and vertical measurements may give incorrect results.

---

|                         |  |
|-------------------------|--|
| <b>Group</b>            | Waveform Transfer  |
| <b>Syntax</b>           | WFMinpre:YOff <NR3><br>WFMinpre:YOff?  |
| <b>Related Commands</b> | <a href="#">WFMinpre:BYT_Nr</a> , <a href="#">WFMinpre:YMUlt</a> , <a href="#">WFMinpre:YOff?</a>  |
| <b>Arguments</b>        | <NR3> is the vertical offset in digitizing levels.   |
| <b>Examples</b>         | WFMINPRE:YOFF 50 specifies that the zero reference point for the incoming waveform is 50 digitizing levels (2 divisions, for 8-bit data) above the center of the data range. |

WFMINPRE:YOFF? might return :WFMINPRE:YOFF 25 indicating that the vertical position of the incoming waveform in digitizing levels.

## WFMinpre:YUNit

Sets or returns the vertical units of the incoming waveform.

Supported units are: %, /Hz, A, A/A, A/V, A/W, A/dB, A/s, AA, AW, AdB, As, B, Hz, IRE, S/s, V, V/A, V/V, V/W, V/dB, V/s, VV, VW, VdB, Volts, Vs, W, W/A, W/V, W/W, W/dB, W/s, WA, WV, WW, WdB, Ws, dB, dB/A, dB/V, dB/W, dB/dB, dBA, dBV, dBW, dBdB, day, degrees, div, hr, min, ohms, percent, s

**Group** Waveform Transfer

**Syntax** WFMinpre:YUNit <QString>  
WFMinpre:YUNit?

**Related Commands** [WFMinpre:YUNit?](#)

**Arguments** <QString> contains a maximum of three alpha characters that represent the vertical unit of measure for the incoming waveform.

**Examples** WFMINPRE:YUNIT? might return :WFMINPRE:YUNIT "V" indicating that the vertical units for the incoming waveform are volts.

WFMINPRE:YUNIT "A" specifies that the vertical units for the incoming waveform are Amperes.

## WFMinpre:YZEro

Sets or returns the vertical offset of the incoming waveform in units specified by WFMinpre:YUNit. Variations in this number are analogous to changing the vertical offset of the waveform.

YMUlt, YOFF, and YZEro are used to convert waveform record values to YUNit values using the following formula (where dl is the data level; curve\_in\_dl is a data point in CURVe):

$$\text{value\_in\_units} = ((\text{curve\_in\_dl} - \text{YOFF\_in\_dl}) * \text{YMUlt}) + \text{YZEro\_in\_units}$$

---

**NOTE.** For a given waveform record, YMUlt, YOFF, and YZEro have to be a consistent set, otherwise vertical cursor readouts and vertical measurements may give incorrect results.

---

|                         |   |
|-------------------------|---|
| <b>Group</b>            | Waveform Transfer   |
| <b>Syntax</b>           | WFMInpre:YZero <NR3><br>WFMInpre:YZero?   |
| <b>Related Commands</b> | <a href="#">WFMInpre:YUNit</a> , <a href="#">WFMOutpre:YZero?</a>   |
| <b>Arguments</b>        | <NR3> is the offset in YUNits.  |
| <b>Examples</b>         | <p>WFMINPRE:YZERO 1.5E+0 specifies that the zero reference point for the incoming waveform is 1.5 V below the center of the data range (given that WFMInpre:YUNit is set to V).</p> <p>WFMINPRE:YZERO? might return :WFMINPRE:YZero 7.5000E-6 indicating that the zero reference for the incoming waveform is 7.5 <math>\mu</math>V below the center of the data range (given that WFMInpre:YUNit is set to V).</p> |

## WFMOutpre? (Query Only)

Returns waveform transmission and formatting parameters for the waveform specified by [DATA:SOURce](#) command. If the waveform specified by the [DATA:SOURce](#) command is not displayed, the oscilloscope returns only the waveform transmission parameters (BYT\_Nr, BIT\_Nr, ENCDg, BN\_Fmt, BYT\_Or).

|                 |  |
|-----------------|--|
| <b>Group</b>    | Waveform Transfer  |
| <b>Syntax</b>   | WFMOutpre?   |
| <b>Examples</b> | <p>WFMOUTPRE? ? might return the waveform formatting data as:<br/>:WFMOUTPRE:BYT_NR 2;BIT_NR 16;ENCDG ASCII;BN_FMT RI;BYT_OR MSB;WFID "Ch1, DC coupling, 100.0mV/div, 4.000us/div, 10000 points, Sample mode";NR_PT 10000;PT_FMT Y;XUNIT "s";XINCR 4.0000E-9;XZERO - 20.0000E-6;PT_OFF 0;YUNIT "V";YMULT 15.6250E-6;YOFF : "6.4000E+3;YZERO 0.0000</p> |

## WFMOutpre:BIT\_Nr

Sets and returns the number of bits per waveform point that outgoing waveforms contain, as specified by the [DATA:SOURce](#) command. Changing the value of



[WFMOutpre:BIT\\_Nr](#) also changes the values of [WFMImpre:FILTERFreq](#) and [DATA:WIDTH](#).

**Group** Waveform Transfer

**Syntax** `WFMOutpre:BIT_Nr <NR1>`  
`WFMOutpre:BIT_Nr?`

**Related Commands** [DATA:SOURce](#), [DATA:WIDTH](#), [WFMOutpre:BN\\_Fmt](#)

**Arguments** <NR1> is the number of bits per data point and can be 8 or 16.

**Examples** `WFMOUTPRE:BIT_NR 16` sets the number of bits per waveform point to 16 for outgoing waveforms.

`WFMOUTPRE:BIT_NR?` might return `:WFMOUTPRE:BIT_NR 8` indicating that outgoing waveforms use 8 bits per waveform point.

## WFMOutpre:BN\_Fmt

Sets or returns the format of binary data for outgoing waveforms specified by the [DATA:SOURce](#) command. Changing the value of [WFMOutpre:BN\\_Fmt](#) also changes the value of [DATA:ENCdg](#).

**Group** Waveform Transfer

**Syntax** `WFMOutpre:BN_Fmt {RI|RP}`  
`WFMOutpre:BN_Fmt?`

**Related Commands** [DATA:ENCdg](#), [DATA:SOURce](#)

**Arguments** RI specifies signed integer data point representation.  
 RP specifies positive integer data point representation.

**Examples** `WFMOUTPRE:BN_FMT RP` specifies that outgoing waveform data will be in positive integer format.

`WFMOUTPRE:BN_FMT?` might return `:WFMOUTPRE:BN_FMT RI` indicating that the outgoing waveform data is currently in signed integer format.

## WFMOutpre:BYT\_Nr

Sets or returns the data width for the outgoing waveform specified by the [DATA:SOURce](#) command. Changing [WFMOutpre:BYT\\_Nr](#) also changes [WFMOutpre:BIT\\_Nr](#) and [DATA:WIDth](#).

**Group** Waveform Transfer

**Syntax** `WFMOutpre:BYT_Nr <NR1>`  
`WFMOutpre:BYT_Nr?`

**Related Commands** [DATA:SOURce](#), [DATA:WIDth](#), [WFMOutpre:BIT\\_Nr](#)

**Arguments** <NR1> is the number of bytes per data point and can be 1 or 2.

**Examples** `WFMOUTPRE:BYT_NR 1` sets the number of bytes per outgoing waveform data point to 1, which is the default setting.

`WFMOUTPRE:BYT_NR?` might return `:WFMOUTPRE:BYT_NR 2` indicating that there are 2 bytes per outgoing waveform data point.

## WFMOutpre:BYT\_Or

Sets or returns which byte of binary waveform data is transmitted first, during a waveform data transfer, when data points require than one byte. This specification only has meaning when [WFMOutpre:ENCdg](#) is set to BIN and [WFMOutpre:BYT\\_Nr](#) is 2. Changing [WFMInpre:FILTERFreq](#) also changes [DATA:ENCdg](#) (if [DATA:ENCdg](#) is not ASCII).

**Group** Waveform Transfer

**Syntax** `WFMOutpre:BYT_Or {LSB|MSB}`  
`WFMOutpre:BYT_Or?`

**Related Commands** [WFMOutpre:ENCdg](#), [WFMOutpre:BYT\\_Nr](#)

**Arguments** LSB specifies that the least significant byte will be transmitted first.

MSB specifies that the most significant byte will be transmitted first.

- Examples** WFMOUTPRE:BYT\_OR MSB sets the most significant outgoing byte of waveform data to be transmitted first.
- WFMOUTPRE:BYT\_OR? might return :WFMOUTPRE:BYT\_OR LSB indicating that the least significant data byte will be transmitted first.

## WFMOutpre:COMPosition

Controls or indicates the type of data used by the [CURVe?](#) query.

**Group** Waveform Transfer

**Syntax** WFMOutpre:COMPosition  
{COMPOSITE\_YT|COMPOSITE\_ENV|SINGULAR\_YT}  
WFMOutpre:COMPosition?

- Arguments** COMPOSITE\_YT uses the sample part of the composite waveform.  
COMPOSITE\_ENV uses the peak-detect part of the composite waveform.  
SINGULAR\_YT uses the sample part of the singular waveform.

- Examples** WFMOUTPRE:COMPOSITION? might return :WFMOUTPRE:COMPOSITION SINGULAR\_YT indicating that the singular data type is used by the [CURVe?](#) query.
- WFMOUTPRE:COMPOSITIONCOMPOSITE\_YT sets the data type used by the [CURVe?](#) query.

## WFMOutpre:FILTERFreq? (Query Only)

Returns the FilterVu low pass filter frequency applied to the LRL waveform record of the source waveform specified by [DATA:SOURce](#). If the [DATA:RESolution](#) is set to FULL, then this is the filter frequency applied to the full resolution (LRL) waveform. If the [DATA:RESolution](#) is set to REDUced, then this is the filter frequency applied to the reduced resolution (thumbnail) waveform.

**Group** Waveform Transfer

**Syntax** WFMOutpre:FILTERFreq?

**Examples** WFMOUTPRE:FILTERFREQ? might return :WFMOUTPRE:FILTERFREQ 200000000 indicating that the FilterVu frequency for the outgoing waveform is 200 MHz.

## WFMOutpre:RECOrdlength? (Query Only)

Returns the record length for the source waveform as specified by the [DATA:SOURce](#) command.

**Group** Waveform Transfer

**Syntax** WFMOutpre:RECOrdlength?

**Examples** WFMOUTPRE:RECORDLENGTH? might return WFMOUTPRE:RECORDLENGTH 6250 indicating that 6250 is the source waveform record length.

## WFMOutpre:ENCdg

Sets and queries the type of encoding for outgoing waveforms.

**Group** Waveform Transfer

**Syntax** WFMOutpre:ENCdg {ASCIi|BINary}  
WFMOutpre:ENCdg?

**Related Commands** [DATA:ENCdg](#), [WFMOutpre:BYT\\_Nr](#), [WFMInpre:FILTERFreq](#), [WFMOutpre:BIT\\_Nr](#), [WFMOutpre:BN\\_Fmt](#)

**Arguments** ASCIi specifies that the outgoing data is to be in ASCII format. Waveforms will be sent as <NR1> numbers.

BINary specifies that outgoing data is to be in a binary format whose further specification is determined by [WFMOutpre:BYT\\_Nr](#), [WFMOutpre:BIT\\_Nr](#), [WFMOutpre:BN\\_Fmt](#) and [WFMInpre:FILTERFreq](#).

**Examples** WFMOUTPRE:ENCDG? might return :WFMOUTPRE:ENCDG BIN indicating that outgoing waveform data will be sent in binary format.

WFMOUTPRE:ENCDG ASC specifies that the outgoing waveform data will be sent in ASCII format.

## WFMOutpre:NR\_Pt? (Query Only)

Returns the number of points for the [DATA:SOURce](#) waveform that will be transmitted in response to a [CURVe?](#) query. The query command will time out and an error will be generated if the waveform specified by [DATA:SOURce](#) is not turned on.

**Group** Waveform Transfer

**Syntax** WFMOutpre:NR\_Pt?

**Related Commands** [CURVe](#), [DATA](#), [DATA:START](#), [DATA:STOP](#), [SAVe:WAVEform](#), [SAVe:WAVEform:FILEFormat](#), [WFMInpre:NR\\_Pt](#)

**Examples** WFMOUTPRE:NR\_PT? might return :WFMOUTPRE:NR\_PT 10000 indicating that there are 10000 data points to be sent.

## WFMOutpre:PT\_Fmt? (Query Only)

Returns the point format for the outgoing waveform specified by the [DATA:SOURce](#) command. Returned values are either ENV, which indicates envelope mode format in which the data is returned as a series of min/max pairs, or Y, which indicates normal waveform points. The query command will time out and an error will be generated if the waveform specified by [DATA:SOURce](#) is not turned on.

**Group** Waveform Transfer

**Syntax** WFMOutpre:PT\_Fmt?

**Related Commands** [CURVe](#), [DATA:SOURce](#)

**Examples** WFMOUTPRE:PT\_FMT? might return :WFMOutpre:PT\_Fmt ENV indicating that the waveform data is a series of min-max pairs.

## WFMOutpre:PT\_Off? (Query Only)

The set form of this command is ignored. The query form always returns a 0, if the waveform specified by [DATA:SOURce](#) is on or displayed. If the waveform

is not displayed, the query form generates an error and returns event code 2244. This command is for compatibility with other Tektronix oscilloscopes.

**Group** Waveform Transfer

**Syntax** WFMOutpre:PT\_Off?

**Related Commands** [DATA:SOURce](#)

**Arguments** Arguments are ignored.

**Examples** WFMOUTPRE:PT\_OFF? might return WFMOUTPRE:PT\_OFF 0 indicating that the waveform specified by DATA:SOURCE is on or displayed.

## WFMOutpre:PT\_ORder? (Query Only)

This query is for compatibility with other Tektronix oscilloscopes and always returns LINEAR.

**Group** Waveform Transfer

**Syntax** WFMOutpre:PT\_Order?

**Related Commands** [DATA:SOURce](#)

**Examples** WFMOUTPRE:PT\_ORDER? returns :WFMOUTPRE:PT\_ORDER LINEAR.

## WFMOutpre:WFId? (Query Only)

Returns a string describing several aspects of the acquisition parameters for the waveform specified by the [DATA:SOURce](#) command. The query command will time out and an error will be generated if the waveform specified by [DATA:SOURce](#) is not turned on.

**Group** Waveform Transfer

**Syntax** WFMOutpre:WFId?

**Related Commands**    [DATA:SOURce](#)

**Returns**    <QString> comprises the following comma-separated fields documented in the tables below:

**Table 2-42: Waveform Suffixes**

| Field            | Description  | Examples   |
|------------------|--|--|
| Source           | The source identification string as it appears in the front-panel scale factor readouts.   | "CH1-4"<br>"Math1"<br>"Ref1-2"   |
| Coupling         | A string describing the vertical coupling of the waveform (the Source1 waveform in the case of Dual Waveform Math).  | "AC coupling"<br>"DC coupling"<br>"GND coupling"   |
| Vert Scale       | A string containing the vertical scale factor of the unzoomed waveform. The numeric portion will always be four digits. The examples cover all known internal units.   | "100.0 mV/div"<br>"20.00 dB/div"<br>"45.00 deg/div"<br>"785.4 mrad/div"<br>"500.0 $\mu$ Vs/div"<br>"10.00 kV/s/div"<br>"200.0 mV/div"<br>"50.00 unk/div" |
| Horiz Scale      | A string containing the horizontal scale factor of the unzoomed waveform. The numeric portion will always be four digits. The examples cover all known internal units. | "100 ms/div"<br>"10.00 kHz/div"  |
| Record Length    | A string containing the number of waveform points available in the entire record. The numeric portion is given as an integer.  | "1000 points"<br>"1000000 points"  |
| Acquisition Mode | A string describing the mode used to acquire the waveform.   | "Sample mode"<br>"Pk Detect mode"<br>"Envelope mode"<br>"Average mode"   |

**Examples**    WFMOUTPRE:WFID? might return :WFMOUTPRE:WFID "Ch1, DC coupling,100.0mVolts/div,500.0 $\mu$ s/div, 1000 points, Sample mode"

## WFMOutpre:XINcr? (Query Only)

Returns the horizontal point spacing in units of WFMOutpre:XUNit for the waveform specified by the [DATA:SOURce](#) command. This value corresponds to the sampling interval. The query command will time out and an error will be generated if the waveform specified by [DATA:SOURce](#) is not turned on.

**Group** Waveform Transfer

**Syntax** WFMOUtpre:XINcr?

**Related Commands** [DATA:SOUrce](#), [WFMOUtpre:XUNit?](#)

**Examples** WFMOUtpre:XINcr? might return :WFMOUtpre:XINcr 10.0000E-6 indicating that the horizontal sampling interval is 10 µs/point.

## WFMOUtpre:XUNit? (Query Only)

Returns the horizontal units for the waveform specified by the [DATA:SOUrce](#) command. The query command will time out and an error will be generated if the waveform specified by [DATA:SOUrce](#) is not turned on.

**Group** Waveform Transfer

**Syntax** WFMOUtpre:XUNit?

**Related Commands** [DATA:SOUrce](#)

**Examples** WFMOUtpre:XUNit? might return :WFMOUtpre:XUNit "HZ" indicating that the horizontal units for the waveform are in Hertz.

## WFMOUtpre:XZEro? (Query Only)

Returns the time coordinate of the first point in the outgoing waveform.

This value is in units of [WFMOUtpre:XUNit?](#). The query command will time out and an error will be generated if the waveform specified by [DATA:SOUrce](#) is not turned on.

**Group** Waveform Transfer

**Syntax** WFMOUtpre:XZEro?

**Related Commands** [DATA:SOUrce](#), [WFMOUtpre:XUNit?](#)



**Examples** WFMOUTPRE:XZERO? might return :WFMOUTPRE:XZERO 5.6300E-9 indicating that the trigger occurred 5.63 ns before the first sample in the waveform record.

## WFMOutpre:YMUlt? (Query Only)

Returns the vertical scale factor per digitizing level in units specified by WFMOutpre:YUNit for the waveform specified by the [DATA:SOURce](#) command. The query command will time out and an error is generated if the waveform specified by [DATA:SOURce](#) is not turned on.

See the description of [WFMInpre:YMUlt](#) to see how this scale factor is used to convert waveform sample values to volts.

**Group** Waveform Transfer

**Syntax** WFMOutpre:YMUlt?

**Related Commands** [DATA:SOURce](#), [WFMInpre:YMUlt](#)

**Examples** WFMOUTPRE:YMULT? might return :WFMOUTPRE:YMULT 4.0000E-3 indicating that the vertical scale for the corresponding waveform is 100 mV/div (for 8-bit waveform data).

## WFMOutpre:YOFf? (Query Only)

Returns the vertical position in digitizing levels for the waveform specified by the [DATA:SOURce](#) command. The query command will time out and an error will be generated if the waveform specified by [DATA:SOURce](#) is not turned on.

See the description of [WFMInpre:YOFf](#) to see how this position is used to convert waveform sample values to volts.

**Group** Waveform Transfer

**Syntax** WFMOutpre:YOFf?

**Related Commands** [DATA:SOURce](#), [WFMOutpre:BYT\\_Nr](#)

**Examples** WFMOUTPRE:YOFF? might return :WFMOUTPRE:YOFF -50.0000E+0 indicating that the position indicator for the waveform was 50 digitizing levels (2 divisions) below center screen (for 8-bit waveform data).

## WFMOutpre:YUNit? (Query Only)

Returns the vertical units for the waveform specified by the [DATA:SOURce](#) command. The query command will time out and an error will be generated if the waveform specified by [DATA:SOURce](#) is not turned on.

**Group** Waveform Transfer

**Syntax** WFMOutpre:YUNit?

**Related Commands** [DATA:SOURce](#)

**Examples** WFMOUTPRE:YUNIT? might return :WFMOUTPRE:YUNIT "dB" indicating that the vertical units for the waveform are measured in decibels.

## WFMOutpre:YZEro? (Query Only)

Returns the vertical offset in units specified by [WFMOutpre:YUNit?](#) for the waveform specified by the [DATA:SOURce](#) command. The query command will time out and an error will be generated if the waveform specified by [DATA:SOURce](#) is not turned on.

See the description of [WFMInpre:YZEro](#) to see how this offset is used to convert waveform sample values to volts.

**Group** Waveform Transfer

**Syntax** WFMOutpre:YZEro?

**Related Commands** [DATA:SOURce](#), [WFMOutpre:YUNit?](#)

**Examples** WFMOUTPRE:YZERO? might return :WFMOUTPRE:YZERO -100.0000E-3 indicating that vertical offset is set to -100 mV.

## ZOOM? (Query Only)

Returns the current vertical and horizontal positioning and scaling of the display.

**Group** Zoom

**Syntax** ZOOM?

**Examples** ZOOM? might return :ZOOM:MODE 1;GRATICULE:SIZE 80;SPLIT EIGHTYTWENTY;;:ZOOM:ZOOM1:STATE 1;SCALE 400.0000E-12;POSITION 46.8986;FACTOR 50.0000E+3;HORIZONTAL:POSITION 46.8986;SCALE 400.0000E-12

## ZOOM{:MODE|:STATE}

Turns Zoom mode on or off. The Zoom mode query returns the current state of Zoom mode. This command is equivalent to pressing the zoom button located on the front panel.

**Group** Zoom

**Syntax** ZOOM{:MODE|:STATE} {ON|OFF|<NR1>}  
ZOOM{:MODE|:STATE}

**Arguments** ON turns on Zoom mode.  
OFF turns off Zoom mode.  
<NR1> = 0 turns off Zoom mode; any other value turns on Zoom mode.

**Examples** ZOOM:MODE OFF turns off Zoom mode.  
ZOOM:MODE? might return :ZOOM:MODE 1 indicating that Zoom mode is currently turned on.

## ZOOM:ZOOM1? (Query Only)

Returns the current horizontal positioning and scaling of the display.

**Group** Zoom

**Syntax** ZOOM:ZOOM1?

**Examples** ZOOM:ZOOM1? might return :ZOOM:ZOOM1:STATE  
1;SCALE 400.0000E-12;POSITION 46.8986;FACTOR  
50.0000E+3;HORIZONTAL:POSITION 46.8986;SCALE 400.0000E-12.

## ZOOM:ZOOM1:FACTOR? (Query Only)

Returns the zoom factor of a particular zoom box.

**Group** Zoom

**Syntax** ZOOM:ZOOM1:FACTOR?

**Returns** <NR1> is the zoom factor of a zoom box.

## ZOOM:ZOOM1:HORIZONTAL:POSITION

Sets or returns the horizontal position for the specified zoom.

**Group** Zoom

**Syntax** ZOOM:ZOOM1:HORIZONTAL:POSITION <NR3>  
ZOOM:ZOOM1:HORIZONTAL:POSITION?

**Arguments** <NR3> is a value from 0 to 100.00 and is the percent of the upper window that is to the left of screen center, when the zoom factor is 1× or greater.

**Examples** ZOOM:ZOOM1:HORIZONTAL:POSITION 50 sets the Zoom1 reference pointer at 50% of acquired waveform.

ZOOM:ZOOM1:HORIZONTAL:POSITION? might return  
:ZOOM1:HORIZONTAL:POSITION 50.0000 indicating that the Zoom1  
reference pointer is currently set at 50% of acquired waveform.

## ZOOM:ZOOM1:HORIZONTAL:SCALE

Sets or returns the zoom horizontal scale for the specified zoom.

|                  |  |
|------------------|--|
| <b>Group</b>     | Zoom   |
| <b>Syntax</b>    | ZOOM:ZOOM1:HORIZONTAL:SCALE <NR3><br>ZOOM:ZOOM1:HORIZONTAL:SCALE?  |
| <b>Arguments</b> | <NR3> is the amount of expansion in the horizontal direction and ranges from 1.0E-3 to 5.0.  |
| <b>Examples</b>  | ZOOM:ZOOM1:HORIZONTAL:SCALE 5 sets the horizontal scale to 5 seconds.<br>ZOOM:ZOOM2:HORIZONTAL:SCALE? might return<br>:ZOOM2:HORIZONTAL:SCALE 1, indicating that the horizontal scale is 1 second. |

## ZOOM:ZOOM1:POSITION

Sets or returns the horizontal position for the specified zoom.

|                  |   |
|------------------|---|
| <b>Group</b>     | Zoom  |
| <b>Syntax</b>    | ZOOM:ZOOM1:POSITION <NR3><br>ZOOM:ZOOM1:POSITION?   |
| <b>Arguments</b> | <NR3> is a value from 0 to 100.00 and is the percent of the upper window that is to the left of screen center, when the zoom factor is 1× or greater. |

## ZOOM:ZOOM1:SCALE

Sets or returns the zoom horizontal scale for the specified zoom.

|                  |   |
|------------------|---|
| <b>Group</b>     | Zoom  |
| <b>Syntax</b>    | ZOOM:ZOOM1:SCALE <NR3><br>ZOOM:ZOOM1:SCALE?   |
| <b>Arguments</b> | <NR3> is the amount of expansion in the horizontal direction and ranges from 1.0E-3 to 5.0. |

## ZOOM:ZOOM1:STATE

Sets or returns the specified zoom on or off, where x is the integer 1 representing the single zoom window.

**Group** Zoom

**Syntax** ZOOM:ZOOM1:STATE {ON|OFF|<NR1>}  
ZOOM:ZOOM1:STATE?

**Arguments** ON turns Zoom 1 on.  
OFF turns Zoom 1 off.  
<NR1> = 0 disables the specified zoom; any other value enables the specified zoom.

**Examples** ZOOM:ZOOM1:STATE ON turns Zoom1 on.  
ZOOM:ZOOM1:STATE? might return :ZOOM:ZOOM1:STATE 1 indicating that Zoom1 is on.

# Status and Events

The oscilloscope provides a status and event reporting system for the Ethernet, GPIB (with the TEK-USB-488 Adapter), and USB interfaces. This system informs you of certain significant events that occur within the oscilloscope.

The oscilloscope status handling system consists of five 8-bit registers and two queues for each interface. The remaining Status subtopics describe these registers and components. They also explain how the event handling system operates.

## Registers

### Overview

The registers in the event handling system fall into two functional groups:

- Status Registers contain information about the status of the oscilloscope. They include the Standard Event Status Register (SESR).
- Enable Registers determine whether selected types of events are reported to the Status Registers and the Event Queue. They include the Device Event Status Enable Register (DESER), the Event Status Enable Register (ESER), and the Service Request Enable Register (SRER).

### Status Registers

The Standard Event Status Register (SESR) and the Status Byte Register (SBR) record certain types of events that may occur while the oscilloscope is in use. IEEE Std 488.2-1987 defines these registers.

Each bit in a Status Register records a particular type of event, such as an execution error or message available. When an event of a given type occurs, the oscilloscope sets the bit that represents that type of event to a value of one. (You can disable bits so that they ignore events and remain at zero. See Enable Registers). Reading the status registers tells you what types of events have occurred.

**The Standard Event Status Register (SESR).** The SESR records eight types of events that can occur within the oscilloscope. Use the \*ESR? query to read the SESR register. Reading the register clears the bits of the register so that the register can accumulate information about new events.

---

**NOTE.** *TekVISA applications use SESR bit 6 to respond to any of several events, including some front panel actions.*

---

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| PON | URQ | CME | EXE | DDE | QYE | RQC | OPC |

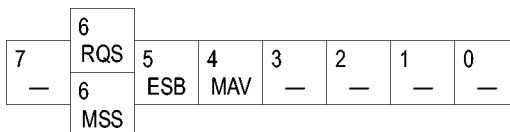
**Figure 3-1: The Standard Event Status Register (SESR)**

**Table 3-1: SESR Bit Functions**

| Bit     | Function |  |
|---------|----------|--|
| 7 (MSB) | PON      | Power On. Shows that the oscilloscope was powered on. On completion, the diagnostic self tests also set this bit.                                |
| 6       | URQ      | User Request. Indicates that an application event has occurred. *See note.   |
| 5       | CME      | Command Error. Shows that an error occurred while the oscilloscope was parsing a command or query.   |
| 4       | EXE      | Execution Error. Shows that an error executing a command or query.   |
| 3       | DDE      | Device Error. Shows that a device error occurred.  |
| 2       | QYE      | Query Error. Either an attempt was made to read the Output Queue when no data was present or pending, or that data in the Output Queue was lost. |
| 1       | RQC      | Request Control. This is not used.   |
| 0 (LSB) | OPC      | operation complete. Shows that the operation is complete. This bit is set when all pending operations complete following an *OPC command.        |

**The Status Byte Register (SBR).** Records whether output is available in the Output Queue, whether the oscilloscope requests service, and whether the SESR has recorded any events.

Use a Serial Poll or the \*STB? query to read the contents of the SBR. The bits in the SBR are set and cleared depending on the contents of the SESR, the Event Status Enable Register (ESER), and the Output Queue. When you use a Serial Poll to obtain the SBR, bit 6 is the RQS bit. When you use the \*STB? query to obtain the SBR, bit 6 is the MSS bit. Reading the SBR does not clear the bits.



**Figure 3-2: The Status Byte Register (SBR)**

**Table 3-2: SBR Bit Functions**

| Bit     | Function |  |
|---------|----------|--|
| 7 (MSB) | —        | Not used.  |
| 6       | RQS      | Request Service. Obtained from a serial poll. Shows that the oscilloscope requests service from the GPIB controller. |
| 6       | MSS      | Master Status Summary. Obtained from *STB? query. Summarizes the ESB and MAV bits in the SBR.                        |
| 5       | ESB      | Event Status Bit. Shows that status is enabled and present in the SESR.  |



**Table 3-2: SBR Bit Functions (cont.)**

| Bit | Function   |
|-----|--|
| 4   | MAV Message Available. Shows that output is available in the Output Queue. |
| 3   | ———— Not used.   |
| 2   | ———— Not used.   |
| 1-0 | ———— Not used.   |

### Enable Registers

DESER, ESER, and SRER allow you to select which events are reported to the Status Registers and the Event Queue. Each Enable Register acts as a filter to a Status Register (the DESER also acts as a filter to the Event Queue) and can prevent information from being recorded in the register or queue.

Each bit in an Enable Register corresponds to a bit in the Status Register it controls. In order for an event to be reported to a bit in the Status Register, the corresponding bit in the Enable Register must be set to one. If the bit in the Enable Register is set to zero, the event is not recorded.

Various commands set the bits in the Enable Registers. The Enable Registers and the commands used to set them are described below.

**The Device Event Status Enable Register (DESER).** This register controls which types of events are reported to the SESR and the Event Queue. The bits in the DESER correspond to those in the SESR.

Use the DESE command to enable and disable the bits in the DESER. Use the DESE? query to read the DESER.

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| PON | URQ | CME | EXE | DDE | QYE | RQC | OPC |

**Figure 3-3: The Device Event Status Enable Register (DESER)**

**The Event Status Enable Register (ESER).** This register controls which types of events are summarized by the Event Status Bit (ESB) in the SBR. Use the \*ESE command to set the bits in the ESER. Use the \*ESE? query to read it.

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
| PON | URQ | CME | EXE | DDE | QYE | RQC | OPC |

**Figure 3-4: The Event Status Enable Register (ESER)**

**The Service Request Enable Register (SRER).** This register controls which bits in the SBR generate a Service Request and are summarized by the Master Status Summary (MSS) bit.

Use the \*SRE command to set the SRER. Use the \*SRE? query to read the register. The RQS bit remains set to one until either the Status Byte Register is read with a Serial Poll or the MSS bit changes back to a zero.

|   |   |     |     |   |   |   |   |
|---|---|-----|-----|---|---|---|---|
| 7 | 6 | 5   | 4   | 3 | 2 | 1 | 0 |
| — | — | ESB | MAV | — | — | — | — |

**Figure 3-5: The Service Request Enable Register (SRER)**

**\*PSC Command** The \*PSC command controls the Enable Registers contents at power-on. Sending \*PSC 1 sets the Enable Registers at power on as follows:

- DESER 255 (equivalent to a DESe 255 command)
- ESER 0 (equivalent to an \*ESE 0 command)
- SRER 0 (equivalent to an \*SRE 0 command)

Sending \*PSC 0 lets the Enable Registers maintain their values in nonvolatile memory through a power cycle.

---

**NOTE.** To enable the PON (Power On) event to generate a Service Request, send \*PSC 0, use the DESe and \*ESE commands to enable PON in the DESER and ESER, and use the \*SRE command to enable bit 5 in the SRER. Subsequent power-on cycles will generate a Service Request.

---

## Queues

The \*PSC command controls the Enable Registers contents at power-on. Sending \*PSC 1 sets the Enable Registers at power on as follows:

**Output Queue** The oscilloscope stores query responses in the Output Queue and empties this queue each time it receives a new command or query message after an <EOM>. The controller must read a query response before it sends the next command (or query) or it will lose responses to earlier queries.




---

**CAUTION.** When a controller sends a query, an <EOM>, and a second query, the oscilloscope normally clears the first response and outputs the second while reporting a Query Error (QYE bit in the ESER) to indicate the lost response. A fast controller, however, may receive a part or all of the first response as well. To avoid this situation, the controller should always read the response immediately after sending any terminated query message or send a DCL (Device Clear) before sending the second query.

---

**Event Queue**

The Event Queue stores detailed information on up to 33 events. If than 32 events stack up in the Event Queue, the 32nd event is replaced by event code 350, "Queue Overflow."

Read the Event Queue with the `EVENT?` query (which returns only the event number), with the `EVMSG?` query (which returns the event number and a text description of the event), or with the `ALLEV?` query (which returns all the event numbers along with a description of the event). Reading an event removes it from the queue.

Before reading an event from the Event Queue, you must use the `*ESR?` query to read the summary of the event from the SESR. This makes the events summarized by the `*ESR?` read available to the `EVENT?` and `EVMSG?` queries, and empties the SESR.

Reading the SESR erases any events that were summarized by previous `*ESR?` reads but not read from the Event Queue. Events that follow an `*ESR?` read are put in the Event Queue but are not available until `*ESR?` is used again.

## Event Handling Sequence

The figure below shows how to use the status and event handling system. In the explanation that follows, numbers in parentheses refer to numbers in the figure.

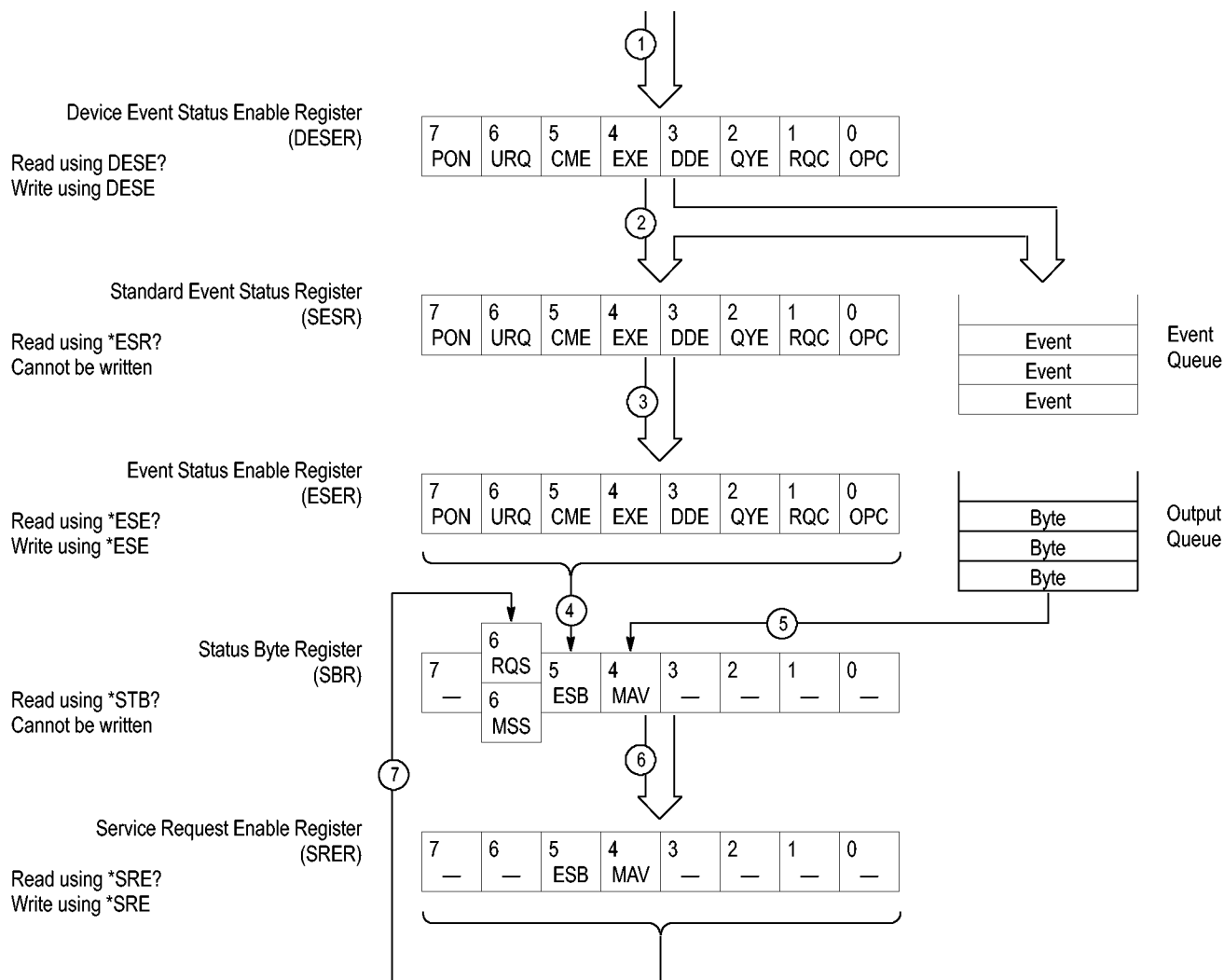


Figure 3-6: Status and Event Handling Process

When an event occurs, a signal is sent to the DESER (1). If that type of event is enabled in the DESER (that is, if the bit for that event type is set to 1), the appropriate bit in the SESR is set to one, and the event is recorded in the Event Queue (2). If the corresponding bit in the ESER is also enabled (3), then the ESB bit in the SBR is set to one (4).

When output is sent to the Output Queue, the MAV bit in the SBR is set to one (5).

When a bit in the SBR is set to one and the corresponding bit in the SRER is enabled (6), the MSS bit in the SBR is set to one and a service request is generated (7).

## Synchronization Methods

**Overview** Although most commands are completed almost immediately after being received by the oscilloscope, some commands start a process that requires time. For example, once a single sequence acquisition command is executed, depending upon the applied signals and trigger settings, it may take an intermediate amount of time before the acquisition is complete. Rather than remain idle while the operation is in process, the oscilloscope will continue processing other commands. This means that some operations will not be completed in the order that they were sent.

Sometimes the result of an operation depends on the result of an earlier operation. A first operation must complete before the next one is processed. The oscilloscope status and event reporting system is designed to accommodate this process.

The Operation Complete (OPC) bit of the Standard Event Status Register (SESR) can be programmed to indicate when certain oscilloscope operations have completed and, by setting the Event Status Enable Register (ESER) to report OPC in the Event Status Bit (ESB) of the Status Byte Register (SBR) and setting the Service Request Enable Register (SRER) to generate service request upon a positive transition of the ESB, a service request (SRQ) interrupt can be generated when certain operations complete as described in this section.

The following oscilloscope operations can generate an OPC:

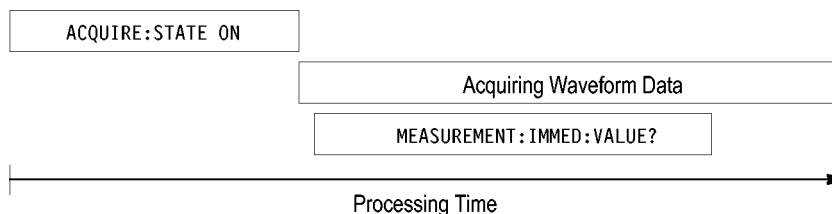
- :ACquire:STATE <non-zero nr1> | ON | only when in single sequence acquisition mode
- :CALibrate:FACTory START
  - :CALibrate:FACTory CONTinue
  - :CALibrate:FACTory PREVIOUS
- :HARDCopy
  - :HARDCopy START
- :DIAg:STATE EXECute
- :SAVe:IMAGe <file as quoted string>
- :SAVe:SETUp <file as quoted string>
- :RECALL:SETUp <file as quoted string>
- :SAVe:WAVEform <file as quoted string>
- :RECALL:WAVEform <file as quoted string>
- :CH<x>:PRObe:DEGAUss EXECute
  - :AUXin:PRObe:DEGAUss EXECute
- TEKSecure

For example, a typical application might involve acquiring a single-sequence waveform and then taking a measurement on the acquired waveform. You could use the following command sequence to do this:

```

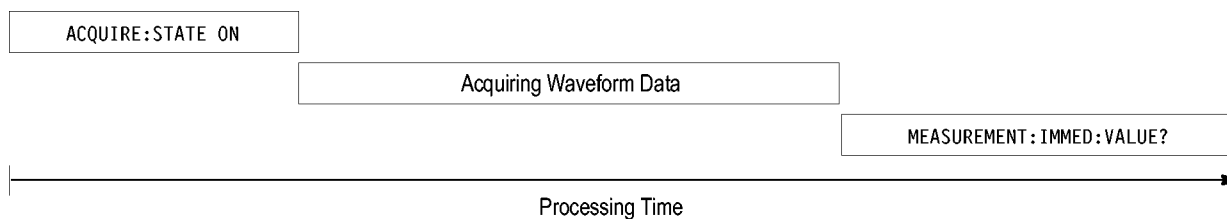
/** Set up conditional acquisition **/
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 1000
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER SEQUENCE
/** Acquire waveform data **/
ACQUIRE:STATE ON
/** Set up the measurement parameters **/
MEASUREMENT:IMMED:TYPE AMPLITUDE
MEASUREMENT:IMMED:SOURCE CH1
/** Take amplitude measurement **/
MEASUREMENT:MEAS1:VALUE
    
```

The acquisition of the waveform requires extended processing time. It may not finish before the oscilloscope takes an amplitude measurement (see the following figure). This can result in an incorrect amplitude value.



**Figure 3-7: Command Processing Without Using Synchronization**

To ensure the oscilloscope completes waveform acquisition before taking the measurement on the acquired data, you can synchronize the program.



**Figure 3-8: Processing Sequence With Synchronization**

You can use four commands to synchronize the operation of the oscilloscope with your application program: \*WAI, BUSY, \*OPC, and \*OPC

**Using the \*WAI Command**

The \*WAI command forces completion of previous commands that generate an OPC message. No commands after the \*WAI are processed before the OPC message(s) are generated

The same command sequence using the \*WAI command for synchronization looks like this:

```

/* Set up conditional acquisition */
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 1000
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER SEQUENCE
/* Acquire waveform data */
ACQUIRE:STATE ON
/* Set up the measurement parameters */
MEASUREMENT:IMMED:TYPE AMPLITUDE
MEASUREMENT:IMMED:SOURCE CH1
/* wait until the acquisition is complete before taking
the measurement*/
*/
*WAI
/* Take amplitude measurement */
MEASUREMENT:IMMED:VALUE

```

The controller can continue to write commands to the input buffer of the oscilloscope, but the commands will not be processed by the oscilloscope until all in-process OPC operations are complete. If the input buffer becomes full, the controller will be unable to write commands to the buffer. This can cause a time-out.

### Using the BUSY Query

The BUSY? query allows you to find out whether the oscilloscope is busy processing a command that has an extended processing time such as single-sequence acquisition.

The same command sequence, using the BUSY? query for synchronization, looks like this:

```

/* Set up conditional acquisition */
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 1000
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER SEQUENCE
/* Acquire waveform data */
ACQUIRE:STATE ON
/* Set up the measurement parameters */
MEASUREMENT:IMMED:TYPE AMPLITUDE
MEASUREMENT:IMMED:SOURCE CH1
/* wait until the acquisition is complete before taking
the measurement */
while BUSY keep looping

```

```

/* Take amplitude measurement */
MEASUREMENT:IMMED:VALUE

```

This sequence lets you create your own wait loop rather than using the \*WAI command. The BUSY? query helps you avoid time-outs caused by writing too many commands to the input buffer. The controller is still tied up though, and the repeated BUSY? query will result in bus traffic.

### Using the \*OPC Command

If the corresponding status registers are enabled, the \*OPC command sets the OPC bit in the Standard Event Status Register (SESR) when an operation is complete. You achieve synchronization by using this command with either a serial poll or service request handler.

**Serial Poll Method:** Enable the OPC bit in the Device Event Status Enable Register (DESER) and the Event Status Enable Register (ESER) using the DESE and \*ESE commands.

When the operation is complete, the OPC bit in the Standard Event Status Register (SESR) will be enabled and the Event Status Bit (ESB) in the Status Byte Register will be enabled.

The same command sequence using the \*OPC command for synchronization with serial polling looks like this:

```

/* Set up conditional acquisition */
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 1000
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER SEQUENCE
/* Enable the status registers */
DESE 1
*ESE 1
*SRE 0
/* Acquire waveform data */
ACQUIRE:STATE ON
/* Set up the measurement parameters */
MEASUREMENT:IMMED:TYPE AMPLITUDE
MEASUREMENT:IMMED:SOURCE CH1
/* wait until the acquisition is complete before taking the
measurement.*/
*OPC
while serial poll = 0, keep looping
/* Take amplitude measurement */
MEASUREMENT:IMMED:VALUE

```

This technique requires less bus traffic than did looping on BUSY.



**Service Request Method:** Enable the OPC bit in the Device Event Status Enable Register (DESER) and the Event Status Enable Register (ESER) using the DESE and \*ESE commands.

You can also enable service requests by setting the ESB bit in the Service Request Enable Register (SRER) using the \*SRE command. When the operation is complete, the oscilloscope will generate a Service Request.

The same command sequence using the \*OPC command for synchronization looks like this

```
/* Set up conditional acquisition */
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 1000
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER SEQUENCE
/* Enable the status registers */
DESE 1
*ESE 1
*SRE 32
/* Acquire waveform data */
ACQUIRE:STATE ON
/* Set up the measurement parameters */
MEASUREMENT:IMMED:TYPE AMPLITUDE
MEASUREMENT:IMMED:SOURCE CH1
/* wait until the acquisition is complete before taking the
measurement*/
*OPC
```

The program can now do different tasks such as talk to other devices. The SRQ, when it comes, interrupts those tasks and returns control to this task.

```
/* Take amplitude measurement */
MEASUREMENT:IMMED:VALUE
```

This technique is efficient but requires sophisticated programming.

### Using the \*OPC? Query

The \*OPC? query places a 1 in the Output Queue once an operation that generates an OPC message is complete. A time out could occur if you try to read the output queue before there is any data in it.

The same command sequence using the \*OPC? query for synchronization looks like this:

```
/* Set up conditional acquisition */
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 1000
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER SEQUENCE
```

```

/* Acquire waveform data */
ACQUIRE:STATE ON
/* Set up the measurement parameters */
MEASUREMENT:IMMED:TYPE AMPLITUDE
MEASUREMENT:IMMED:SOURCE CH1
/* wait until the acquisition is complete before taking the
measurement*/
*OPC

Wait for read from Output Queue.
/* Take amplitude measurement */
MEASUREMENT:IMMED:VALUE
    
```

This is the simplest approach. It requires no status handling or loops. However, you must set the controller time-out for longer than the acquisition operation.

**Messages**

The information contained in the topic tabs above covers all the programming interface messages the oscilloscope generates in response to commands and queries.

For most messages, a secondary message from the oscilloscope gives detail about the cause of the error or the meaning of the message. This message is part of the message string and is separated from the main message by a semicolon.

Each message is the result of an event. Each type of event sets a specific bit in the SESR and is controlled by the equivalent bit in the DESER. Thus, each message is associated with a specific SESR bit. In the message tables, the associated SESR bit is specified in the table title, with exceptions noted with the error message text.

**No Event**

The following table shows the messages when the system has no events or status to report. These have no associated SESR bit.

**Table 3-3: No Event Messages**

| Code | Message                                       |
|------|---|
| 0    | No events to report; queue empty              |
| 1    | No events to report; new events pending *ESR? |

**Command Error**

The following table shows the command error messages generated by improper syntax. Check that the command is properly formed and that it follows the rules in the section on command Syntax.

**Table 3-4: Command Error Messages (CME Bit 5)**

| Code | Message           |
|------|-------------------|
| 100  | Command error     |
| 101  | Invalid character |

**Table 3-4: Command Error Messages (CME Bit 5) (cont.)**

| <b>Code</b> | <b>Message</b>               |
|-------------|------------------------------|
| 102         | Syntax error                 |
| 103         | Invalid separator            |
| 104         | Data type error              |
| 105         | GET not allowed              |
| 108         | Parameter not allowed        |
| 109         | Missing parameter            |
| 110         | Command header error         |
| 112         | Program mnemonic too long    |
| 113         | Undefined header             |
| 120         | Numeric data error           |
| 121         | Invalid character in numeric |
| 123         | Exponent too large           |
| 124         | Too many digits              |
| 130         | Suffix error                 |
| 131         | Invalid suffix               |
| 134         | Suffix too long              |
| 140         | Character data error         |
| 141         | Invalid character data       |
| 144         | Character data too long      |
| 150         | String data error            |
| 151         | Invalid string data          |
| 152         | String data too long         |
| 160         | Block data error             |
| 161         | Invalid block data           |
| 170         | Command expression error     |
| 171         | Invalid expression           |

**Execution Error**

The following table lists the execution errors that are detected during execution of a command.

**Table 3-5: Execution Error Messages (EXE Bit 4)**

| <b>Code</b> | <b>Message</b>          |
|-------------|-------------------------|
| 200         | Execution error         |
| 221         | Settings conflict       |
| 222         | Data out of range       |
| 224         | Illegal parameter value |

Table 3-5: Execution Error Messages (EXE Bit 4) (cont.)

| Code | Message   |
|------|---|
| 241  | Hardware missing  |
| 250  | Mass storage error                                      |
| 251  | Missing mass storage                                    |
| 252  | Missing media   |
| 253  | Corrupt media   |
| 254  | Media full  |
| 255  | Directory full  |
| 256  | File name not found                                     |
| 257  | File name error   |
| 258  | Media protected   |
| 259  | File name too long                                      |
| 270  | Hardcopy error  |
| 271  | Hardcopy device not responding                          |
| 272  | Hardcopy is busy  |
| 273  | Hardcopy aborted  |
| 274  | Hardcopy configuration error                            |
| 280  | Program error   |
| 282  | Insufficient network printer information                |
| 283  | Network printer not responding                          |
| 284  | Network printer server not responding                   |
| 286  | Program runtime error                                   |
| 287  | Print server not found                                  |
| 2200 | Measurement error, Measurement system error             |
| 2201 | Measurement error, Zero period                          |
| 2202 | Measurement error, No period, second waveform           |
| 2203 | Measurement error, No period, second waveform           |
| 2204 | Measurement error, Low amplitude, second waveform       |
| 2205 | Measurement error, Low amplitude, second waveform       |
| 2206 | Measurement error, Invalid gate                         |
| 2207 | Measurement error, Measurement overflow                 |
| 2208 | Measurement error, No backwards Mid Ref crossing        |
| 2209 | Measurement error, No second Mid Ref crossing           |
| 2210 | Measurement error, No Mid Ref crossing, second waveform |
| 2211 | Measurement error, No backwards Mid Ref crossing        |
| 2212 | Measurement error, No negative crossing                 |
| 2213 | Measurement error, No positive crossing                 |
| 2214 | Measurement error, No crossing, target waveform         |

**Table 3-5: Execution Error Messages (EXE Bit 4) (cont.)**

| <b>Code</b> | <b>Message</b>   |
|-------------|--|
| 2215        | Measurement error, No crossing, second waveform                |
| 2216        | Measurement error, No crossing, target waveform                |
| 2217        | Measurement error, Constant waveform                           |
| 2219        | Measurement error, No valid edge - No arm sample               |
| 2220        | Measurement error, No valid edge - No arm cross                |
| 2221        | Measurement error, No valid edge - No trigger cross            |
| 2222        | Measurement error, No valid edge - No second cross             |
| 2223        | Measurement error, Waveform mismatch                           |
| 2224        | Measurement error, WAIT calculating                            |
| 2225        | Measurement error, No waveform to measure                      |
| 2226        | Measurement error, Null Waveform                               |
| 2227        | Measurement error, Positive and Negative Clipping              |
| 2228        | Measurement error, Positive Clipping                           |
| 2229        | Measurement error, Negative Clipping                           |
| 2230        | Measurement error, High Ref < Low Ref                          |
| 2231        | Measurement error, No statistics available                     |
| 2233        | Requested waveform is temporarily unavailable                  |
| 2235        | Math error, invalid math description                           |
| 2240        | Invalid password   |
| 2241        | Waveform requested is invalid                                  |
| 2244        | Source waveform is not active                                  |
| 2245        | Saveref error, selected channel is turned off                  |
| 2250        | Reference error, the reference waveform file is invalid        |
| 2253        | Reference error, too many points received                      |
| 2254        | Reference error, too few points received                       |
| 2259        | File too big   |
| 2270        | Alias error  |
| 2271        | Alias syntax error   |
| 2273        | Illegal alias label  |
| 2276        | Alias expansion error  |
| 2277        | Alias redefinition not allowed                                 |
| 2278        | Alias header not found   |
| 2285        | TekSecure(R) Pass  |
| 2286        | TekSecure(R) Fail  |
| 2500        | Setup error, file does not look like a setup file              |
| 2501        | Setup warning, could not recall all values from external setup |
| 2620        | Mask error, too few points received                            |

**Table 3-5: Execution Error Messages (EXE Bit 4) (cont.)**

| <b>Code</b> | <b>Message</b>     |
|-------------|--------------------|
| 2760        | Mark limit reached |
| 2761        | No mark present    |
| 2762        | Search copy failed |

**Device Error** The following table lists the device errors that can occur during oscilloscope operation. These errors may indicate that the oscilloscope needs repair.

**Table 3-6: Device Error Messages (DDE Bit 3)**

| <b>Code</b> | <b>Message</b>          |
|-------------|-------------------------|
| 310         | System error            |
| 311         | Memory error            |
| 312         | PUD memory lost         |
| 314         | Save/recall memory lost |

**System Event** The following table lists the system event messages. These messages are generated whenever certain system conditions occur.

**Table 3-7: System Event Messages**

| <b>Code</b> | <b>Message</b>   |
|-------------|--|
| 400         | Query event  |
| 401         | Power on (PON bit 7 set)                                     |
| 402         | Operation complete (OPC bit 0 set)                           |
| 403         | User request (URQ bit 6 set)                                 |
| 404         | Power fail (DDE bit 3 set)                                   |
| 405         | Request control  |
| 410         | Query INTERRUPTED (QYE bit 2 set)                            |
| 420         | Query UNTERMINATED (QYE bit 2 set)                           |
| 430         | Query DEADLOCKED (QYE bit 2 set)                             |
| 440         | Query UNTERMINATED after indefinite response (QYE bit 2 set) |
| 468         | Knob/Keypad value changed                                    |
| 472         | Application variable changed                                 |

**Execution Warning** The following table lists warning messages that do not interrupt the flow of command execution. They also notify you of a possible unexpected results.

**Table 3-8: Execution Warning Messages (EXE Bit 4)**

| <b>Code</b> | <b>Message</b>                                  |
|-------------|---|
| 528         | Parameter out of range                          |
| 532         | Curve data too long, Curve truncated            |
| 533         | Curve error, Preamble values are inconsistent   |
| 540         | Measurement warning, Uncertain edge             |
| 541         | Measurement warning, Low signal amplitude       |
| 542         | Measurement warning, Unstable histogram         |
| 543         | Measurement warning, Low resolution             |
| 544         | Measurement warning, Uncertain edge             |
| 545         | Measurement warning, Invalid in minmax          |
| 546         | Measurement warning, Need 3 edges               |
| 547         | Measurement warning, Clipping positive/negative |
| 548         | Measurement warning, Clipping positive          |
| 549         | Measurement warning, Clipping negative          |

**Table 3-9: Execution Warning Messages (EXE Bit 4)**

| <b>Code</b> | <b>Message</b>                                  |
|-------------|---|
| 540         | Measurement warning                             |
| 541         | Measurement warning, Low signal amplitude       |
| 542         | Measurement warning, Unstable histogram         |
| 543         | Measurement warning, Low resolution             |
| 544         | Measurement warning, Uncertain edge             |
| 545         | Measurement warning, Invalid min max            |
| 546         | Measurement warning, Need 3 edges               |
| 547         | Measurement warning, Clipping positive/negative |
| 548         | Measurement warning, Clipping positive          |
| 549         | Measurement warning, Clipping negative          |

**Internal Warning**

The following table shows internal errors that indicate an internal fault in the oscilloscope.

**Table 3-10: Internal Warning Messages**

| <b>Code</b> | <b>Message</b>                 |
|-------------|--------------------------------|
| 630         | Internal warning, 50Ω overload |

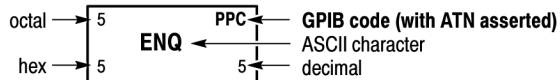




# Appendix A: Character Set

| B7<br>B6<br>B5<br>BITS<br>B4 B3 B2 B1 | 0 0                   |                       | 0 0 1               |                   | 0 1 0                              |       | 0 1 1      |                  | 1 0 0      |   | 1 0 1 |   | 1 1 0      |   | 1 1 1 |   |
|---------------------------------------|-----------------------|-----------------------|---------------------|-------------------|------------------------------------|-------|------------|------------------|------------|---|-------|---|------------|---|-------|---|
|                                       | CONTROL               |                       |                     |                   | NUMBERS<br>SYMBOLS                 |       |            |                  | UPPER CASE |   |       |   | LOWER CASE |   |       |   |
| 0 0 0 0                               | 0 NUL                 | 20 DLE                | 40 SP               | 60 0              | 100 @                              | 120 P | 140 ' SA0  | 160 p SA16       | 0          | 0 | 0     | 0 | 0          | 0 | 0     | 0 |
| 0 0 0 1                               | 1 SOH                 | 21 DC1                | 41 !                | 61 1              | 101 A                              | 121 Q | 141 a SA1  | 161 q SA17       | 1          | 1 | 1     | 1 | 1          | 1 | 1     | 1 |
| 0 0 1 0                               | 2 STX                 | 22 DC2                | 42 "                | 62 2              | 102 B                              | 122 R | 142 b SA2  | 162 r SA18       | 2          | 2 | 2     | 2 | 2          | 2 | 2     | 2 |
| 0 0 1 1                               | 3 ETX                 | 23 DC3                | 43 #                | 63 3              | 103 C                              | 123 S | 143 c SA3  | 163 s SA19       | 3          | 3 | 3     | 3 | 3          | 3 | 3     | 3 |
| 0 1 0 0                               | 4 EOT                 | 24 DC4                | 44 \$               | 64 4              | 104 D                              | 124 T | 144 d SA4  | 164 t SA20       | 4          | 4 | 4     | 4 | 4          | 4 | 4     | 4 |
| 0 1 0 1                               | 5 ENQ                 | 25 NAK                | 45 %                | 65 5              | 105 E                              | 125 U | 145 e SA5  | 165 u SA21       | 5          | 5 | 5     | 5 | 5          | 5 | 5     | 5 |
| 0 1 1 0                               | 6 ACK                 | 26 SYN                | 46 &                | 66 6              | 106 F                              | 126 V | 146 f SA6  | 166 v SA22       | 6          | 6 | 6     | 6 | 6          | 6 | 6     | 6 |
| 0 1 1 1                               | 7 BEL                 | 27 ETB                | 47 ' LA7            | 67 7              | 107 G                              | 127 W | 147 g SA7  | 167 w SA23       | 7          | 7 | 7     | 7 | 7          | 7 | 7     | 7 |
| 1 0 0 0                               | 8 BS                  | 30 CAN                | 50 ( LA8            | 70 8              | 110 H                              | 130 X | 150 h SA8  | 170 x SA24       | 8          | 8 | 8     | 8 | 8          | 8 | 8     | 8 |
| 1 0 0 1                               | 9 HT                  | 31 EM                 | 51 ) LA9            | 71 9              | 111 I                              | 131 Y | 151 i SA9  | 171 y SA25       | 9          | 9 | 9     | 9 | 9          | 9 | 9     | 9 |
| 1 0 1 0                               | A LF                  | 32 SUB                | 52 * LA10           | 72 :              | 112 J                              | 132 Z | 152 j SA10 | 172 z SA26       | A          | A | A     | A | A          | A | A     | A |
| 1 0 1 1                               | B VT                  | 33 ESC                | 53 + LA11           | 73 ; LA27         | 113 K                              | 133 [ | 153 k SA11 | 173 { SA27       | B          | B | B     | B | B          | B | B     | B |
| 1 1 0 0                               | C FF                  | 34 FS                 | 54 , LA12           | 74 < LA28         | 114 L                              | 134 \ | 154 l SA12 | 174   SA28       | C          | C | C     | C | C          | C | C     | C |
| 1 1 0 1                               | D CR                  | 35 GS                 | 55 - LA13           | 75 = LA29         | 115 M                              | 135 ] | 155 m SA13 | 175 } SA29       | D          | D | D     | D | D          | D | D     | D |
| 1 1 1 0                               | E SO                  | 36 RS                 | 56 . LA14           | 76 > LA30         | 116 N                              | 136 ^ | 156 n SA14 | 176 ~ SA30       | E          | E | E     | E | E          | E | E     | E |
| 1 1 1 1                               | F SI                  | 37 US                 | 57 / LA15           | 77 ? UNL          | 117 O                              | 137 - | 157 o SA15 | 177 RUBOUT (DEL) | F          | F | F     | F | F          | F | F     | F |
|                                       | ADDRESSED<br>COMMANDS | UNIVERSAL<br>COMMANDS | LISTEN<br>ADDRESSES | TALK<br>ADDRESSES | SECONDARY ADDRESSES<br>OR COMMANDS |       |            |                  |            |   |       |   |            |   |       |   |

## KEY



## Tektronix

REF: ANSI STD X3.4-1977  
IEEE STD 488.1-1987  
ISO STD 646-2973



---

# Appendix B: Waveform Data in DPO/MSO2000 Series Instruments

An analog channel of an DPO/MSO2000 series instrument has a sample waveform and a peak detect waveform being acquired simultaneously. This combination waveform is referred to as a “composite” waveform. In some cases, for example when averaging is turned on, or for math waveforms, a composite waveform is not available. In these cases, the waveform is termed “singular”, and is available only in sample mode.

## Full Versus Reduced Resolution Waveforms

The DPO/MSO2000 series introduces the concept of reduced resolution waveforms available for curve queries. The DPO/MSO2000 oscilloscopes acquire channel data into “full” resolution records that are nominally 1 million (1M) or 100,000 (100K) points in size (the acquisition record length); these are termed “full resolution”, or LRL (long record length) waveforms. These LRL records are “shrunk” (decimated) down to “thumbnail”, “short”, or “reduced resolution” waveforms; these are what appear on the oscilloscope’s display screen. During decimation, a low-pass filter operation may be applied to the LRL sample data that results in a filtered thumbnail waveform. In general, both full and reduced resolution waveforms can be queried from the oscilloscope. The size of the reduced resolution waveform varies from 800 to 6250 points, depending on instrument settings.

For applications which need to query waveform data rapidly for the purpose of displaying it, reduced resolution waveform data will provide a faster display update than a full resolution waveform data.

## Nominal Versus Actual Record Length

The DPO/MSO2000 series also introduces the concept of nominal versus actual acquisition record length. Like other oscilloscopes, the acquisition record length can be set using the [HORizontal:RECOrdlength](#) command. However, the actual number of points in the waveform record may differ from the nominal record length established by the [HORizontal:RECOrdlength](#) setting, depending on the [HORizontal:SCALE](#) in effect at the time of the acquisition. The valid values for setting acquisition record length are 1,000,000 (1M) and 100,000 (100K) points. Depending on the horizontal scale in effect at the time of the acquisition, a nominal 1M point record may be comprised of 1.0M or 1.25M (1,250,000) points and a nominal 100K point record may be comprised of 100K or 125K (125,000) points. The actual number of points in the waveform record for the source waveform (specified by the [DATA:SOURce](#) command) may be obtained using the [WFMOutpre:RECOrdlength?](#) query.

## Composition of Waveform Data

The DPO/MSO2000 series introduces the concept of composite waveform data for analog channels. Sample data is always available for acquired analog channel waveforms. Composite waveforms are available in **SAMPLE** acquisition mode but are not available in **AVERAGE** acquisition mode. Peak data may be available

depending on the instrument bandwidth, acquisition mode, nominal record length, and the horizontal scale settings.

- Sample waveform data which may be a component of a composite waveform is referred to as **COMPOSITE\_YT** (vertical units versus time) data.
- Peak detect waveform data which may be a component of a composite waveform is referred to as **COMPOSITE\_ENV** (envelope or peak detect min/max pairs versus time) data.
- Analog channel waveforms acquired in **AVERAGE** acquisition mode and **MATH** waveforms are not composite waveforms and are termed **SINGULAR\_YT** which indicates that the waveform is a single waveform composed of data points representing vertical units versus time. The composition of digital waveform data (:DATA:SOURce D0 through D15, DIGital) is **SINGULAR\_YT** only.

The following table lists the various waveform data compositions:

| Composition                               | Description  |
|---|--|
| COMPOSITE_YT<br>(analog channels)         | This is sample waveform data in vertical units versus time. Its native width is 8-bits and is written as 8-bit data in saved ISF waveform files and reference waveforms. The data can be queried as 1- or 2-byte data in <a href="#">CURVe?</a> queries by setting the data width to 1 or 2.                                 |
| COMPOSITE_ENV<br>(analog channels)        | This is min/max pair waveform data with the minimum value preceding the maximum value. Its native width is 8-bits and is written as 8-bit data in saved ISF waveform files and reference waveforms. The data can be queried as 1- or 2-byte data in <a href="#">CURVe?</a> queries by setting the data width to 1 or 2.      |
| SINGULAR_YT<br>(analog and math channels) | This is 16-bit waveform data which originates as averaged acquired waveform data or math data. It is written to saved ISF waveform files and reference waveforms as 16- bit data. The data can be queried as 1-or 2- byte data in <a href="#">CURVe?</a> queries by setting the data width to 1 or 2.                        |
| SINGULAR_YT<br>(digital channels)         | This is sample waveform data in binary vertical units (0 or 1) versus time. Its native width is 8-bits for individual digital channels (D0 through D15) and 32 bits for the digital collection (D0 through D15, digital representation of CH1 through CH4). The only resolution available for digital waveform data is FULL. |

The available waveform compositions, which can be obtained using the [CURVe?](#) or [WAVFrm?](#) queries, can be determined using the [DATA:COMPosition:AVAILable?](#) query. This query returns the available waveform compositions for the last acquired waveform for the source specified by the [DATA:SOURce](#) command. The data returned by the [DATA:COMPosition:AVAILable?](#) query may be one of the following:

- COMPOSITE\_YT
- COMPOSITE\_YT, COMPOSITE\_ENV
- SINGULAR\_YT

The composition of the waveform data to be queried using the [CURVe?](#) or [WAVFrm?](#) queries is set using the [DATA:COMPosition](#) or [WFMOupre:COMPosition](#) command. The compositions which can be set are:

- COMPOSITE\_YT
- COMPOSITE\_ENV
- SINGULAR\_YT

If the composition set is not available with the current acquisition and horizontal settings, no response is returned for the [CURVe?](#) or [WAVFrm?](#) query and an event is set.

For example, if [COMPOSITE\\_ENV](#) data is not available and a [CURVe?](#) query is sent with the composition set to [COMPOSITE\\_ENV](#), no data will be returned and an error event is set:

```
> :DATA:COMPOSITION:AVAILABLE?
COMPOSITE_YT
> DATA:COMPOSITION?
COMPOSITE_YT
> DATA:COMPOSITION
COMPOSITE_ENV
> *ESR? 0 > CURVE? {no data returned}
> *ESR?
20
> EVMSG?
221,"Settings conflict; Envelope waveform not available;
CURVE?" >
```

The composition of the waveform data to be sent to the instrument for transfer to the reference waveform memory using the [CURVe](#) command is set using the [WFMinpre:COMPosition](#) command. The compositions which can be set are:

- COMPOSITE\_YT
- COMPOSITE\_ENV
- SINGULAR\_YT

### Filtering of Waveform Data with FilterVu

The FilterVu feature uses digital signal processing (DSP) techniques to apply low pass filtering to the acquired waveform data for the analog channels. The FilterVu feature provides filtering of the displayed waveform so that you can see your signal without the effects of noise. This is the reduced resolution waveform. A side effect of filtering the reduced resolution waveform is that the full resolution waveform may be subjected to filtering to achieve the desired filtering of the reduced resolution waveform. The filtering frequency of the full resolution waveform is not under your direct control. It is a function of the horizontal and acquisition settings and the selected FilterVu frequency. The 3 dB cutoff frequency for this filtering may be different for the LRL record and the thumbnail

record. The actual filter frequency applied to the acquired (LRL) and displayed (thumbnail) waveforms are available in the waveform preamble:

- [WFMOutpre:FILTERFreq?](#)

This query returns the 3 dB low pass filter cutoff frequency for:

- The acquired (LRL) waveform when the [DATA:RESOLUTION](#) is set to FULL.
- The displayed (thumbnail) waveform when the [DATA:RESOLUTION](#) is set to REDUCED.

When waveforms are sent to the instrument using the [CURVe](#) command or recalled from ISF files, the filtering frequency is obtained from the waveform preamble:

- [WFMinpre:FILTERFreq](#)

This command sets the 3 dB low pass filter cutoff frequency for the waveform being transferred to the instrument's reference waveform memory.

Only FULL resolution waveforms may be sent to the instrument for storage and display in reference waveform memory.

The filtering of reduced resolution waveforms is controlled using the [FILTERVu:FREQuency](#) command. The available low pass frequencies depends on the instrument bandwidth and the horizontal settings (scale and record length) and may be queried for the last acquired waveform, (as specified by the [DATA:SOURce](#) command) using the [FILTERVu:FREQuency:AVAILable?](#) query. This query returns a comma separated list of available FilterVu frequencies in decreasing order for the last acquired source waveform . For example, on a 100 MHz bandwidth instrument at a full resolution record length of 1 M points and a horizontal scale of 20 microseconds per division, the query returns these values which apply to the reduced resolution (displayed) waveform: 100000000, 75000000, 42000000, 21000000, 11000000, 5500000, and 2200000.

The low pass filter frequency in effect can be queried using the [FILTERVu:FREQuency?](#) query.

### FilterVu Controls for the Displayed Waveform

The FilterVu feature can be turned on or off using the [DISPlay:GLITCh](#) command and the display intensity of the glitch capture waveform can be controlled using the [DISPlay:INTENSITY:GLITCh](#) command.

### Querying Acquisition Peak Detect Data from the Instrument

1. Use the [DATA:COMPosition:AVAILable?](#) query to determine if peak detect data (COMPOSITE\_ENV) is available. This query returns a list of waveform data types that are available from the instrument under the current instrument settings:
  - COMPOSITE\_YT for sample data
  - COMPOSITE\_ENV for envelope or peak detect data
  - SINGULAR\_YT for sample data
2. If COMPOSITE\_ENV is available, set the data composition to COMPOSITE\_ENV (:DATA:COMPosition COMPOSITE\_ENV). Otherwise, the peak detect data is not available and cannot be queried from the instrument.
3. [CURVe?](#) will now return peak detect data.

### Querying Acquisition YT Sample Data from the Instrument

The sequence used to query sample data is nearly identical to the one used for peak detect data. Set the data composition to COMPOSITE\_YT or SINGULAR\_YT (based on results of the [DATA:COMPosition:AVAILable?](#) query); [CURVe?](#) will now return sample data.

### Selecting Acquisition Record Length

Like other oscilloscopes, the acquisition record length can be set using the [HORizontal:RECOrdlength](#) command. However, the actual number of points in the acquired waveform record may differ from the nominal record length established by the [HORizontal:RECOrdlength](#) setting, depending on the [HORizontal:SCAle](#) in effect at the time of the acquisition. The allowed values for setting the acquisition record length are 1,000,000 (1 million) and 100,000 (100 thousand) points. Depending on the horizontal scale in effect at the time of the acquisition, a nominal 1 million point record may be composed of 1.0 million or 1.25 million points and a nominal 100 thousand point record may be composed of 100 thousand or 125 thousand points. The actual number of points acquired for the source waveform (specified by the [DATA:SOUrce](#) command) may be obtained using the [WFMOutpre:RECOrdlength?](#) query.

For reduced resolution waveforms, the actual record length depends on the FilterVu frequency in effect in addition to the horizontal settings. The reduced resolution record lengths vary with the FilterVu frequency.

High Resolution and Envelope acquisition modes are not supported in DPO/MSO2000 series instruments.

For the digital waveforms (D0 through D15), the actual record length may differ from the nominal record length established by the [HORizontal:RECOrdlength](#) setting, depending on the [HORizontal:SCAle](#) in effect at the time of the acquisition and whether digital waveforms are selected from one or both of the digital pods (whether digital channels are turned on for D0 through D7, and for D8 through D15). For faster horizontal scales, the actual record length may be half of the nominal record length when digital waveforms from both digital pods are

turned on. Use the [WFMOutpre:RECOrdlength?](#) query to determine the actual record length for digital waveforms.

### Extensions to the DATA Commands

Like the DPO4000 series, the DPO2000 series provides the [DATA:WIDTH](#) command that allows you to specify the data word width (in bytes) used when querying data from the instrument or when sending data to the instrument with the [CURVe](#) command. The following are the extensions to data commands:

- [DATA:RESOlution](#)
- [DATA:COMPosition:AVAILable?](#)
- [DATA:COMPosition](#)

### WFMOutpre Parameters

The [WFMOutpre?](#) parameters correspond to the source waveform specified by the [DATA:SOURce](#) command. The following are the extensions to Waveform Outpre commands:

- [WFMOutpre:COMPosition](#)
- [WFMOutpre:FILTErFreq?](#)
- [WFMOutpre:RECOrdlength?](#)

### WFMinpre Parameters

The [WFMinpre?](#) parameters correspond to the destination reference waveform specified by the [DATA:DESTination](#) command. The following are the extensions to Waveform Inpre commands:

- [WFMinpre:FILTErFreq](#)
- [WFMinpre:COMPosition](#)

### FilterVu Frequency Control Commands

The following are the FilterVu commands:

- [FILTERVu:FREQuency](#)
- [FILTERVu:FREQuency:AVAILable?](#)
- [DISplay:GLITCh](#)
- [DISplay:INTENSITy:GLITCh](#)



# Appendix C: Reserved Words

This is a list of reserved words for your instrument. Capital letters identify the required minimum spelling.

|               |             |                |               |
|---------------|-------------|----------------|---------------|
| *CAL          | ADDRANDDATA | BINary         | CMEan         |
| *CLS          | ADDResS     | BITOrder       | COLUMN        |
| *DDT          | ADDress     | BITRate        | COMMAND       |
| *ESE          | ALIAS       | BIT_Nr         | COMPOSITE_ENV |
| *ESR          | ALIAS[      | BLAckmanharris | COMPOSITE_YT  |
| *IDN          | ALL         | BM             | COMPosition   |
| *LRN          | ALLEv       | BMP            | CONDition     |
| *OPC          | ALLFields   | BN_Fmt         | CONFigure     |
| *PSC          | ALLLines    | BOTH           | CONTRol       |
| *PUD          | ALTERNATE   | BOX            | CONTInue      |
| *RCL          | ALWAYS      | BTRIGger       | COPY          |
| *RST          | AMPliTude   | BURSt          | COUNT         |
| *SAV          | AND         | BUS            | COUPLing      |
| *SRE          | ANY         | BUSY           | CPU           |
| *STB          | APPKey      | BY             | CR            |
| *TRG          | AREa        | BYPass         | CRC           |
| *TST          | ASCIi       | BYT_Nr         | CRCHeader     |
| *WAI          | ASCIi       | BYT_Or         | CRCTrailer    |
| 1             | ASCIi       | CALibrate      | CREATE        |
| 1CH           | ASSIgn      | CAN            | CRMs          |
| 7             | ATRIGger    | CANH           | CROSSHair     |
| 8             | AUTO        | CANL           | CURSor        |
| 9             | AUTOSet     | CARD           | CURSors       |
| A             | AUTOZero    | CAREa          | CURVe         |
| A0            | AUX         | CATaLog        | CURrent       |
| A1            | AUXOut      | CH             | CUSTOM        |
| A2            | AUXin       | CH1            | CWD           |
| A3            | AVAILable   | CH2            | CYCLEcount    |
| A4            | AVERage     | CH3            | D             |
| A5            | Auto        | CH4            | D0            |
| A6            | B           | CHanneL        | D1            |
| A7            | B0          | CHEcksum       | D10           |
| A8            | B1          | CLAss          | D11           |
| A9            | B2          | CLEAR          | D12           |
| ABORT         | B3          | CLEARMenu      | D13           |
| ABOrt         | B4          | CLEARSNapshot  | D14           |
| ABSolute      | B5          | CLEAr          | D15           |
| AC            | B6          | CLOCK          | D2            |
| ACKMISS       | B7          | CLOCK          | D3            |
| ACQ           | B8          | CM10BY15       | D4            |
| ACQLENGTH     | B9          | CM13BY18       | D5            |
| ACQuire       | BACKLight   | CM15BY21       | D6            |
| ACTIVeprinter | BACKwards   | CM18BY24       | D7            |
| ADD           | BANdwidth   | CM6BY8         | D8            |
| ADDR10        | BASE        | CM7BY10        | D9            |
| ADDR7         | BDIFFBP     | CM9BY13        | DATA          |

|              |             |             |             |
|--------------|-------------|-------------|-------------|
| DATABits     | ENV         | FRAMetype   | HTTPPort    |
| DATE         | ENvelope    | FRAMING     | I2C         |
| DATEPrint    | EOF         | FRame       | ID          |
| DATA         | EOFTYPE     | FREE        | IDANDDATA   |
| DAll         | EQua1       | FREESpace   | IDFORmat    |
| DB           | EQua1       | FRENch      | IDLETime    |
| DC           | ERROR       | FREQuency   | IDLEtime    |
| DECimal      | ERRTYPE     | FULL        | IDPrint     |
| DEFLT        | ERRlog      | FUL1        | IDentifier  |
| DEFault      | ERRor       | FUNction    | IMAGESize   |
| DEFine       | ETHERnet    | GAIN        | IMAge       |
| DEGAuss      | EVEN        | GATEWay     | IMAge       |
| DEGrees      | EVENT       | GATIng      | IMMed       |
| DELEte       | EVENTS      | GATIng      | IMPedance   |
| DELIMiter    | EVENTtable  | GENeralcall | IN          |
| DELTA        | EVMsg       | GERMan      | IN11BY17    |
| DELtatime    | EVQty       | GLITCh      | IN2P5BY3P25 |
| DELAy        | EXECute     | GND         | IN4BY6      |
| DELAyed      | EXT         | GPIBUsb     | IN8BY10     |
| DELeTe       | EXTended    | GRaticule   | INDBits     |
| DELta        | EXTernal    | GRId        | INDICators  |
| DESE         | FACTOR      | HAGAKIPC    | INDIVidual  |
| DESKew       | FACTory     | HAGAKIPCARD | INDEpendent |
| DESTination  | FAIL        | HALFgrat    | INFINite    |
| DHCPbootp    | FALL        | HAMming     | INIT        |
| DIAG         | FALSe       | HANning     | INKSaver    |
| DIFFerential | FALLing     | HARDCopy    | INPUt       |
| DIGItal      | FASTPHOTO   | HBArS       | INTENSITY   |
| DIGital      | FASter      | HD1080I50   | INTERLAcEd  |
| DIGtal       | FAStest     | HD1080I60   | INTERNA1    |
| DIR          | FFT         | HD1080P24   | INVERTed    |
| DIREction    | FIELD       | HD1080P25   | INVERT      |
| DISabled     | FIFTy       | HD1080PSF24 | INVERTed    |
| DISplay      | FILEFormat  | HD480P60    | INrange     |
| DISplaymode  | FILESystem  | HD720P60    | IO          |
| DNS          | FILTERFreq  | HdTv        | IPADDRESS   |
| DOMAINname   | FILTERVu    | HEADER      | ISCLOCKed   |
| DPOModels    | FINE        | HEAdEr      | ITALian     |
| DRAFT        | FIRSt       | HEIght      | JAPANEse    |
| DUAL         | FIVEdivs    | HERtz       | KOREan      |
| DUMp         | FLAg        | HEXadecima1 | L           |
| DYNAMIc      | FOCUS       | HFRej       | L2          |
| E            | FOLDER      | HIGH        | L4          |
| ECL          | FORCEDRange | HIGHLimit   | LABEL       |
| EDGE         | FORCE       | HIREs       | LABe1       |
| EDGE         | FORMAT      | HISTogram   | LANGUage    |
| EEPROM       | FORMAT      | HIVALue     | LANdScape   |
| EITHer       | FORwards    | HOLDTime    | LARGE       |
| ENAbLe       | FPAnel      | HOLDoff     | LAYout      |
| ENCdg        | FRACTIONal  | HORZ        | LESSEQua1   |
| END          | FRAMEID     | HORizonta1  | LESSLimit   |
| ENET         | FRAMetype   | HPOS        | LESSThan    |
| ENGLISH      | FRAMETypeid | HSmode      | LESSThan    |

|                |               |               |              |
|----------------|---------------|---------------|--------------|
| LETTER         | MOREThan      | PARity        | RATE25K      |
| Leve]          | MOREthan      | PASS          | RATE35K      |
| LF             | MOSI          | PASSword      | RATE50K      |
| LFRej          | MSB           | PATtern       | RATE800K     |
| LIN            | NAME          | PAYLength     | RATIO        |
| LINE           | NAME]         | PAYLoad       | RDELta       |
| LINEAr         | NANd          | PERCent       | READ         |
| LINEPeriod     | NEGative      | PERIod        | READFile     |
| LIST           | NEWpass       | PERSistence   | RECA]l       |
| LOCK           | NEXT          | PERcent       | RECOrdlength |
| LOG            | NEXt          | PHase         | RECTX        |
| LOGIC          | NO            | PHOTO         | RECTY        |
| LOGIC          | NOCARE        | PictBridge    | RECTangular  |
| LOGic          | NOISerej      | PING          | REDUced      |
| LOOP           | NONE          | PK2Pk         | REF          |
| LOW            | NONE          | PLAIN         | REF1         |
| LOWLimit       | NOPARity      | PNG           | REF2         |
| Lowerthreshold | NORMAl        | POLARity      | REFLeve]l    |
| LSB            | NOTCOMPuted   | POLAr         | REM          |
| MAG            | NR1           | POLarity      | REMOte       |
| MAGnivu        | NRMAL         | PORTRait      | REName       |
| MAIN           | NR_Pt         | PORTUguese    | REPEATstart  |
| MAIn           | NTIMES        | POSITION      | RESET        |
| MAIn]          | NTIMes        | POSITion      | RESOlution   |
| MARK           | NTSc          | POSITive      | RESUlT       |
| MARKSINCOLumn  | NULL          | POvershoot    | RESistance   |
| MATH           | NULLFRDynamic | POWERupstatus | RESUlts      |
| MATH1          | NULLFRStatic  | PPULSECount   | RI           |
| MATHVAR        | NUL]l         | PRESS         | RIBinary     |
| MATH[1]        | NUMACq        | PREVIEW       | RISEFall     |
| MAXSamplerate  | NUMAVg        | PREVIEWstate  | RISE         |
| MAXimum        | NUMENTries    | PREVIOUS      | RISing       |
| MEAN           | NUMERIC       | PRINTQual     | RMDir        |
| MEAS           | NUMenv        | PRINTEr       | RMS          |
| MEASurement    | NUMHORZ       | PRODELta      | ROLL100MM    |
| MEDIum         | NUMVERT       | PRODUCT       | ROLL127MM    |
| MEG            | OCCURS        | PRODUCTID     | ROLL210MM    |
| MESSAge        | ODD           | PROGRESSive   | ROLL89MM     |
| METHod         | OFF           | PRObe         | ROM          |
| MID            | OFFSet        | PT_Fmt        | RP           |
| MID2           | ON            | PT_ORder      | RPBinary     |
| MINImum        | ONCE          | PT_Off        | RS232        |
| MINMax         | ONFAIL        | PULSEWIDth    | RS232C       |
| MISO           | OPTion        | PULSEwidth    | RUN          |
| MISOMOSI       | OUT           | PULSe         | RUNSTop      |
| MIXed          | OUTrange      | PULSe         | RUNT         |
| MKDir          | OVERLoad      | PWIDth        | RUNT         |
| MM100BY150     | OWner         | QUALifier     | RUSSian      |
| MM54BY86       | PACKET        | RADIUS        | RWINCLude    |
| MODE           | PAL           | RATDELta      | RX           |
| MODE           | PAPERSize     | RATE15K       | RXDATA       |
| MOREEQual      | PAPERType     | RATE1M        | RXENDPacket  |
| MORELimit      | PARalle]l     | RATE20K       | RXPARity     |

|                   |               |                    |           |
|-------------------|---------------|--------------------|-----------|
| RXStArt           | SOURCE[1]     | THDELta            | VALue     |
| SAMPLERate        | SOURce        | THETA              | VAR       |
| SAMPLEpoint       | SOURce2       | THReshold          | VBARs     |
| SAMple            | SPANish       | THReshold          | VDELTA    |
| SAVe              | SPC           | TIFf               | VENDORID  |
| SCAN              | SPECTra1      | TIME               | VERBoSe   |
| SCAlE             | SPI           | TIME               | VERT      |
| SCLK              | SPREADSheet   | TOTAL              | VERTical  |
| SCREEN            | SPace         | TOTAl              | VIDeo     |
| SCREen            | SRIBinary     | TOTALuptime        | VOLTs     |
| SDATA             | SRPbinary     | TRACk              | WAKEup    |
| SEARCH            | SS            | TRADitionalchinese | WAVEform  |
| SEARCHtotrigger   | STANDard      | TRANSition         | WAVFrm    |
| SECAM             | STANDard      | TRIGger            | WEIghting |
| SECOnds           | START         | TRIGgertosearch    | WFID      |
| SElect            | STARTofframe  | TRUE               | WFMInpre  |
| SElected          | STARTup       | TTL                | WFMOutpre |
| SEQUence          | STARTupnosync | TURN               | WHEN      |
| SERIALnumber      | START         | TWEnty             | WIDth     |
| SERnumber         | STARTbyte     | TWOfifty           | WINDow    |
| SET               | STATE         | TX                 | WRITE     |
| SETHold           | STATE]        | TXDATA             | WRITEFile |
| SETLevel          | STATISTICS    | TXENDPacket        | X         |
| SETTime           | STATUS        | TXPARity           | XDELta    |
| SETUP             | STATE         | TXRX               | XFF       |
| SETUp             | STATIC        | TXSTART            | XINcr     |
| SEVENTYFive       | STATUS        | TYPE               | XUNit     |
| SHOW              | STDdev        | TYPE               | XY        |
| SIGna1            | STOP          | Than               | XZErO     |
| SIMPlifiedchinese | STOPAfter     | UNDO               | Y         |
| SINGULAR_YT       | STRing        | UNEQual            | YDELta    |
| SIZE              | STYle         | UNIts              | YES       |
| SLEEP             | STANDard      | UNLock             | YMUlt     |
| SLEWRate          | SUBNETMask    | UPPerthreshold     | YOFF      |
| SLOWer            | SYNC          | USBDevice          | YT        |
| SLOpe             | SYNCFieLd     | USBTMC             | YUNIts    |
| SMA11             | SYNCFrAmE     | USBTmc             | YUNit     |
| SNAPShot          | SYNcIntervAl  | USE                | YUNits    |
| SNAp              | TEKSecure     | USER               | YZErO     |
| SOF               | TEMPerature   | V1X                | ZOOM      |
| SOURCE            | TERmination   | V2X                | ZOOM      |
| SOURCE2           | TESTnumber    |                    |           |

# Index

## A

ACQUIRE:MAXSAMPLERATE?, 2-57  
ACQUIRE?, 2-57  
ACQUIRE:MODE, 2-57  
ACQUIRE:NUMACQ?, 2-58  
ACQUIRE:NUMAVG, 2-59  
ACQUIRE:STATE, 2-59  
ACQUIRE:STOPAFTER, 2-60  
Acquisition Command Group, 2-11  
Alias Command Group, 2-12  
ALIAS, 2-61  
ALIAS[:STATE], 2-64  
ALIAS:CATALOG?, 2-62  
ALIAS:DEFINE, 2-62  
ALIAS:DELETE, 2-63  
ALIAS:DELETE[:NAME], 2-64  
ALIAS:DELETE:ALL, 2-63  
ALLEV?, 2-64  
AUTOSet, 2-65  
AUTOSet:ENABLE, 2-65  
AUXIN:PROBE:AUTOZERO, 2-66  
AUXIN:PROBE:DEGAUSS:STATE?, 2-68  
AUXIN:PROBE:FORCEDRANGE, 2-68  
AUXIN:PROBE:ID:SERIALNUMBER?, 2-69  
AUXIN:PROBE:ID:TYPE?, 2-69  
AUXIN:PROBE:RESISTANCE?, 2-69  
AUXIN?, 2-66  
AUXIN:PROBE, 2-66  
AUXIN:PROBE:COMMAND, 2-67  
AUXIN:PROBE:DEGAUSS, 2-67  
AUXIN:PROBE:GAIN, 2-68  
AUXIN:PROBE:SIGNAL, 2-70  
AUXIN:PROBE:UNITS?, 2-70

## B

BUS, 2-70  
BUS:B<x>:CAN:BITRATE, 2-71  
BUS:B<x>:CAN:PROBE, 2-71  
BUS:B<x>:CAN:SAMPLEPOINT, 2-71  
BUS:B<x>:CAN:SOURCE, 2-72  
BUS:B<x>:DISPLAY:FORMAT, 2-72  
BUS:B<x>:I2C:ADDRESS:RWINCLUDE, 2-73  
BUS:B<x>:I2C{:CLOCK|:SCLK}:SOURCE, 2-73

BUS:B<x>:I2C{:DATA|:SDATA}:SOURCE, 2-74  
BUS:B<x>:LIN:BITRATE, 2-74  
BUS:B<x>:LIN:IDFORMAT, 2-75  
BUS:B<x>:LIN:POLARITY, 2-75  
BUS:B<x>:LIN:SAMPLEPOINT, 2-76  
BUS:B<x>:LIN:SOURCE, 2-76  
BUS:B<x>:LIN:STANDARD, 2-77  
BUS:B<x>:PARALLEL:BIT<x>:SOURCE, 2-77  
BUS:B<x>:PARALLEL:CLOCK:EDGE, 2-78  
BUS:B<x>:PARALLEL:CLOCK:ISCLOCKED, 2-78  
BUS:B<x>:PARALLEL:CLOCK:SOURCE, 2-78  
BUS:B<x>:PARALLEL:WIDTH, 2-79  
BUS:B<x>:POSITION, 2-79  
BUS:B<x>:RS232C:BITRATE, 2-80  
BUS:B<x>:RS232C:DATABITS, 2-80  
BUS:B<x>:RS232C:DELIMITER, 2-80  
BUS:B<x>:RS232C:DISPLAYMODE, 2-81  
BUS:B<x>:RS232C:PARITY, 2-81  
BUS:B<x>:RS232C:POLARITY, 2-82  
BUS:B<x>:RS232C:RX:SOURCE, 2-82  
BUS:B<x>:RS232C:TX:SOURCE, 2-83  
BUS:B<x>:SPI:BITORDER, 2-86  
BUS:B<x>:SPI:DATA:SIZE, 2-87  
BUS:B<x>:SPI:DATA{:IN|:MISO}:POLARITY, 2-84  
BUS:B<x>:SPI:DATA{:IN|:MISO}:SOURCE, 2-84  
BUS:B<x>:SPI:DATA{:OUT|:MOSI}:  
POLARITY, 2-85  
BUS:B<x>:SPI:DATA{:OUT|:MOSI}:SOURCE, 2-85  
BUS:B<x>:SPI:FRAMING, 2-87  
BUS:B<x>:SPI:IDLETIME, 2-88  
BUS:B<x>:SPI{:CLOCK|:SCLK}:POLARITY, 2-83  
BUS:B<x>:SPI{:CLOCK|:SCLK}:SOURCE, 2-83  
BUS:B<x>:SPI{:SELECT|:SS}:POLARITY, 2-85  
BUS:B<x>:SPI{:SELECT|:SS}:SOURCE, 2-86  
BUS:LOWERTHRESHOLD:CH<x>, 2-89  
BUS:THRESHOLD:CH<x>, 2-90  
BUS:THRESHOLD:D<x>, 2-90  
BUS:UPPERTHRESHOLD:CH<x>, 2-91  
BUS:B<x>:LABEL, 2-74  
BUS:B<x>:STATE, 2-88  
BUS:B<x>:TYPE, 2-89  
BUSY?, 2-91

**C**

\*CAL?, 2-92  
 CALibrate:FACTory:STATus?, 2-93  
 CALibrate:INTERNaL:START, 2-94  
 CALibrate:INTERNaL:STATus?, 2-94  
 CALibrate:RESults:FACTory?, 2-95  
 CALibrate:RESults:SPC?, 2-95  
 CALibrate:TEMPerature?, 2-96  
 CALibrate:FACTory, 2-93  
 CALibrate:INTERNaL, 2-93  
 CALibrate:RESults?, 2-95  
 Calibration and Diagnostic Command Group, 2-16  
 CH<x>:PRObe:AUTOZero, 2-101  
 CH<x>:PRObe:COMMAND, 2-101  
 CH<x>:PRObe:DEGAUss, 2-102  
 CH<x>:PRObe:DEGAUss:STATE?, 2-102  
 CH<x>:PRObe:FORCEDRange, 2-102  
 CH<x>:PRObe:ID:SERnumber?, 2-104  
 CH<x>:PRObe:ID:TYPE?, 2-104  
 CH<x>:PRObe:RESistance?, 2-104  
 CH<x>:PRObe:SIGNAL, 2-105  
 CH<x>:PRObe:UNIts?, 2-105  
 CH<x>:TERmination, 2-106  
 CH<x>?, 2-96  
 CH<x>:BANdwidth, 2-96  
 CH<x>:COUPling, 2-97  
 CH<x>:DESKew, 2-97  
 CH<x>:IMPedance, 2-98  
 CH<x>:INVert, 2-98  
 CH<x>:LABel, 2-99  
 CH<x>:OFFSet, 2-99  
 CH<x>:POSition, 2-100  
 CH<x>:PRObe?, 2-100  
 CH<x>:PRObe:GAIN, 2-103  
 CH<x>:PRObe:ID?, 2-103  
 CH<x>:SCALE, 2-106  
 CH<x>:VOLts, 2-107  
 CH<x>:YUNits, 2-107  
 CLEARMenu, 2-107  
 \*CLS, 2-108  
 Command Groups, 2-11  
 Cursor Command Group, 2-17  
 CURSor:HBArS:POSITION<x>, 2-110  
 CURSor:VBArS:ALTERNATE<x>?, 2-113  
 CURSor:VBArS:HPOS<x>?, 2-113  
 CURSor:VBArS:POSITION<x>, 2-114  
 CURSor:VBArS:VDELTA?, 2-116

CURSor:XY:POLar:RADIUS:DELta?, 2-116  
 CURSor:XY:POLar:RADIUS:POSITION<x>?, 2-117  
 CURSor:XY:POLar:RADIUS:UNIts?, 2-117  
 CURSor:XY:POLar:THETA:DELta?, 2-117  
 CURSor:XY:POLar:THETA:POSITION<x>?, 2-117  
 CURSor:XY:POLar:THETA:UNIts?, 2-117  
 CURSor:XY:PRODUCT:DELta?, 2-118  
 CURSor:XY:PRODUCT:POSITION<x>?, 2-118  
 CURSor:XY:PRODUCT:UNIts?, 2-118  
 CURSor:XY:RATIO:DELta?, 2-118  
 CURSor:XY:RATIO:POSITION<x>?, 2-119  
 CURSor:XY:RATIO:UNIts?, 2-119  
 CURSor:XY:RECTangular:X:DELta?, 2-119  
 CURSor:XY:RECTangular:X:POSITION<x>, 2-119  
 CURSor:XY:RECTangular:X:UNIts?, 2-120  
 CURSor:XY:RECTangular:Y:DELta?, 2-120  
 CURSor:XY:RECTangular:Y:POSITION<x>, 2-120  
 CURSor:XY:RECTangular:Y:UNIts?, 2-120  
 CURSor?, 2-108  
 CURSor:FUNCTion, 2-109  
 CURSor:HBArS?, 2-109  
 CURSor:HBArS:DELTA?, 2-110  
 CURSor:HBArS:UNIts, 2-111  
 CURSor:HBArS:USE, 2-111  
 CURSor:MODE, 2-112  
 CURSor:VBArS?, 2-112  
 CURSor:VBArS:DELTA?, 2-113  
 CURSor:VBArS:UNIts, 2-115  
 CURSor:VBArS:USE, 2-115  
 CURVe, 2-121

**D**

D<x>, 2-142  
 D<x>:LABel, 2-142  
 D<x>:POSition, 2-142  
 D<x>:THREshold, 2-143  
 DATa, 2-122  
 DATa:COMPosition:AVAILable?, 2-124  
 DATa:COMPosition, 2-123  
 DATa:DESTination, 2-125  
 DATa:ENCdg, 2-125  
 DATa:RESOLution, 2-123  
 DATa:SOURce, 2-126  
 DATa:START, 2-128  
 DATa:STOP, 2-129  
 DATa:WIDth, 2-130  
 DATE, 2-130

\*DDT, 2-131  
 DESE, 2-132  
 DIAg:LOOP:OPTion:NTIMes, 2-133  
 DIAg:SElect:<function>, 2-135  
 DIAg:LOOP:OPTion, 2-132  
 DIAg:LOOP:STOP, 2-133  
 DIAg:RESUlt:FLAg?, 2-134  
 DIAg:RESUlt:LOG?, 2-134  
 DIAg:SElect, 2-135  
 DIAg:STATE, 2-136  
 Display Command Group, 2-18  
 DISplay:DIGital:HEIght, 2-137  
 DISplay:INTENSITy:BACKLight, 2-139  
 DISplay:INTENSITy:GLITCh, 2-140  
 DISplay:INTENSITy:GRATicule, 2-140  
 DISplay:INTENSITy:WAVEform, 2-140  
 DISplay?, 2-136  
 DISplay:CLOCK, 2-136  
 DISplay:FORMat, 2-137  
 DISplay:GLITCh, 2-138  
 DISplay:GRATicule, 2-138  
 DISplay:INTENSITy?, 2-139  
 DISplay:PERsistence, 2-141

## E

\*ESE, 2-143  
 \*ESR?, 2-144  
 Ethernet Command Group, 2-19  
 ETHERnet:DNS:IPADdress, 2-145  
 ETHERnet:ENET:ADdress?, 2-146  
 ETHERnet:GATEWay:IPADdress, 2-146  
 ETHERnet:PING:STATUS?, 2-148  
 ETHERnet:DHCPbootp, 2-144  
 ETHERnet:DOMAINname, 2-145  
 ETHERnet:HTTpport, 2-146  
 ETHERnet:IPADdress, 2-147  
 ETHERnet:NAME, 2-147  
 ETHERnet:PASSWord, 2-148  
 ETHERnet:PING, 2-148  
 ETHERnet:SUBNETMask, 2-149  
 EVENT?, 2-149  
 EVMsg?, 2-150  
 EVQty?, 2-150

## F

FACtory, 2-151

File System Command Group, 2-20  
 FILESystem, 2-152  
 FILESystem:FREESpace?, 2-154  
 FILESystem:WRITEFile, 2-157  
 FILESystem:CWD, 2-152  
 FILESystem:DELEte, 2-153  
 FILESystem:DIR?, 2-153  
 FILESystem:FORMat, 2-154  
 FILESystem:MKDir, 2-155  
 FILESystem:READFile, 2-155  
 FILESystem:REName, 2-156  
 FILESystem:RMDir, 2-156  
 FILTERVu:FREQuency:AVAILable?, 2-158  
 FILTERVu:FREQuency, 2-157  
 FPAnel:PRESS, 2-158  
 FPAnel:TURN, 2-160

## G

GPIBUsb:ID?, 2-161

## H

Hard copy Command Group, 2-22  
 HARDCopy, 2-161  
 HARDCopy:INKSaver, 2-162  
 HARDCopy:PREVIEW, 2-162  
 HEADer, 2-163  
 Horizontal Command Group, 2-23  
 HORizontal:ACQLENGTH?, 2-164  
 HORizontal:DELay:MODE, 2-164  
 HORizontal:DELay:TIME, 2-165  
 HORizontal:DIGital:RECOrdlength:MAIN?, 2-165  
 HORizontal:DIGital:SAMPLERate:MAIN?, 2-165  
 HORizontal:PREViewstate?, 2-166  
 HORizontal:RECOrdlength, 2-166  
 HORizontal:RESOLution, 2-167  
 HORizontal:SAMPLERate, 2-167  
 HORizontal:TRIGger:POSition, 2-168  
 HORizontal?, 2-163  
 HORizontal:POSition, 2-166  
 HORizontal:SCAle, 2-167

## I

ID?, 2-168  
 \*IDN?, 2-169

- L**
- LANGUage, 2-169
  - LOCK, 2-169
  - \*LRN?, 2-170
- M**
- Mark Command Group, 2-23
  - MARK, 2-171
  - MARK:SELEcted:FOCUS?, 2-173
  - MARK:SELEcted:MARKSINCOLUMN?, 2-173
  - MARK:SELEcted:OWNer?, 2-173
  - MARK:SELEcted:SOURCE?, 2-174
  - MARK:SELEcted:START?, 2-174
  - MARK:SELEcted:STATe?, 2-174
  - MARK:SELEcted:ZOOM:POSition?, 2-174
  - MARK:CREATE, 2-171
  - MARK:DELEte, 2-172
  - MARK:FREE?, 2-172
  - MARK:SELEcted:END?, 2-173
  - MARK:TOTal?, 2-175
  - Math Command Group, 2-25
  - MATH[1]:HORizontal:POSition, 2-177
  - MATH[1]:HORizontal:SCALE, 2-177
  - MATH[1]:HORizontal:UNIts, 2-178
  - MATH[1]:SPECTral:MAG, 2-178
  - MATH[1]:SPECTral:WINDow, 2-178
  - MATH[1]:VERTical:POSition, 2-179
  - MATH[1]:VERTical:SCALE, 2-180
  - MATH[1]:VERTical:UNIts?, 2-180
  - MATH[1]?, 2-175
  - MATH[1]:DEFine, 2-176
  - MATH[1]:TYPE, 2-179
  - {MATH|MATH1}:LABel, 2-175
  - Measurement Command Group, 2-26
  - MEASUrement:CLEARSNAPSHOT, 2-182
  - MEASUrement:IMMed:DELay:DIRection, 2-183
  - MEASUrement:IMMed:DELay:EDGE<x>, 2-184
  - MEASUrement:IMMed:DELay?, 2-183
  - MEASUrement:IMMed:SOUrce<x>, 2-186
  - MEASUrement:IMMed:SOUrce1, 2-184
  - MEASUrement:IMMed:SOUrce2, 2-185
  - MEASUrement:IMMed:TYPE, 2-186
  - MEASUrement:IMMed:UNIts?, 2-189
  - MEASUrement:IMMed:VALue?, 2-189
  - MEASUrement:INDICators:HORZ<x>?, 2-190
  - MEASUrement:INDICators:NUMHORZ?, 2-191
  - MEASUrement:INDICators:NUMVERT?, 2-191
  - MEASUrement:INDICators:STATE, 2-191
  - MEASUrement:INDICators:VERT<x>?, 2-192
  - MEASUrement:INDICators?, 2-190
  - MEASUrement:MEAS<x>:COUNt?, 2-193
  - MEASUrement:MEAS<x>:DELay:DIRection, 2-193
  - MEASUrement:MEAS<x>:DELay:EDGE<x>, 2-194
  - MEASUrement:MEAS<x>:DELay?, 2-193
  - MEASUrement:MEAS<x>:MAXimum?, 2-194
  - MEASUrement:MEAS<x>:MEAN?, 2-195
  - MEASUrement:MEAS<x>:MINImum?, 2-195
  - MEASUrement:MEAS<x>:SOURCE[1], 2-195
  - MEASUrement:MEAS<x>:SOUrce<x>, 2-197
  - MEASUrement:MEAS<x>:SOUrce2, 2-196
  - MEASUrement:MEAS<x>:STATE, 2-197
  - MEASUrement:MEAS<x>:STDdev?, 2-198
  - MEASUrement:MEAS<x>:TYPE, 2-198
  - MEASUrement:MEAS<x>:UNIts?, 2-201
  - MEASUrement:MEAS<x>:VALue?, 2-201
  - MEASUrement:MEAS<x>?, 2-192
  - MEASUrement:REFLevel:ABSolute:HIGH, 2-203
  - MEASUrement:REFLevel:ABSolute:LOW, 2-203
  - MEASUrement:REFLevel:ABSolute:MID, 2-204
  - MEASUrement:REFLevel:ABSolute:MID<x>, 2-205
  - MEASUrement:REFLevel:ABSolute:MID2, 2-205
  - MEASUrement:REFLevel:METHod, 2-206
  - MEASUrement:REFLevel:PERCent:HIGH, 2-206
  - MEASUrement:REFLevel:PERCent:LOW, 2-207
  - MEASUrement:REFLevel:PERCent:MID, 2-208
  - MEASUrement:REFLevel:PERCent:MID<x>, 2-209
  - MEASUrement:REFLevel:PERCent:MID2, 2-208
  - MEASUrement:REFLevel?, 2-202
  - MEASUrement:SNAPShot, 2-209
  - MEASUrement?, 2-181
  - MEASUrement:GATing, 2-182
  - MEASUrement:IMMed?, 2-182
  - MEASUrement:METHod, 2-202
  - MESSAge, 2-210
  - MESSAge:BOX, 2-210
  - MESSAge:CLEAR, 2-211
  - MESSAge:SHOW, 2-211
  - MESSAge:STATE, 2-214
  - Miscellaneous Command Group, 2-29
- N**
- NEWpass, 2-214



**O**

\*OPC, 2-215

**P**

PASSWord, 2-221

PICTBridge:DATEPrint, 2-217

PICTBridge:IMAGESize, 2-219

PICTBridge:PAPERSize, 2-216

PICTBridge:PAPERType, 2-220

PICTBridge:PRINTQual, 2-220

PICTBridge:DEFault, 2-218

PICTBridge:IDPrint, 2-218

\*PSC, 2-221

\*PUD, 2-222

**R**

\*RCL, 2-222

RECALL:SETUp, 2-223

RECALL:WAVEform, 2-224

REF<x>:HORizontal:DELay:TIME, 2-225

REF<x>:HORizontal:SCALE, 2-225

REF<x>:VERTical:POSition, 2-227

REF<x>:VERTical:SCALE, 2-228

REF<x>?, 2-224

REF<x>:DATE?, 2-224

REF<x>:LABEL, 2-225

REF<x>:POSition, 2-226

REF<x>:SCALE, 2-226

REF<x>:TIME?, 2-227

REM, 2-228

\*RST, 2-229

**S**

\*SAV, 2-229

Save and Recall Command Group, 2-31

SAVe:EVENTtable:BUS<x>, 2-230

SAVe:IMAGe:FILEFormat, 2-231

SAVe:WAVEform:FILEFormat, 2-234

SAVe:WAVEform:GATIng, 2-235

SAVe:WAVEform:SPREADSheet:RESOLution, 2-235

SAVe:ASSIgn:TYPE, 2-230

SAVe:IMAGe, 2-231

SAVe:IMAGe:LAYout, 2-232

SAVe:SETUp, 2-232

SAVe:WAVEform, 2-233

Search Commands Group, 2-33

SEARCH:SEARCH<x>:COPY, 2-239

SEARCH:SEARCH<x>:STATE, 2-239

SEARCH:SEARCH<x>:TOTAL?, 2-239

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN:CONDition, 2-242

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN:DATA:DIRection, 2-242

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN:DATA:QUALifier, 2-243

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN:DATA:SIZE, 2-244

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN:DATA:VALue, 2-244

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN:FRAMEtype, 2-245

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN{:IDentifier|:ADDRes}:MODE, 2-245

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:

CAN{:IDentifier|:ADDRes}:VALue, 2-246

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:

ADDRes:MODE, 2-246

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:

ADDRes:TYPE, 2-246

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:

ADDRes:VALue, 2-247

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:

CONDition, 2-247

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:

DATA:DIRection, 2-248

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:

DATA:SIZE, 2-248

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:I2C:

DATA:VALue, 2-249

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:

CONDition, 2-249

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:

DATA:HIVALue, 2-250

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:

DATA:QUALifier, 2-250

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:

DATA:SIZE, 2-251

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:

DATA:VALue, 2-252

SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:

ERRTYPE, 2-252

- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:LIN:  
IDentifier:VALue, 2-253
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:  
PARallel:VALue, 2-253
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:  
RS232C:CONDition, 2-254
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:  
RS232C:RX:DATA:SIZE, 2-254
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:  
RS232C:RX:DATA:VALue, 2-255
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:  
RS232C:TX:DATA:SIZE, 2-255
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:  
RS232C:TX:DATA:VALue, 2-255
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:  
CONDition, 2-256
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:  
DATA:SIZE, 2-257
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:  
DATA{:MISO|:IN}:VALue, 2-256
- SEARCH:SEARCH<x>:TRIGger:A:BUS:B<x>:SPI:  
DATA{:MOSI|:OUT}:VALue, 2-257
- SEARCH:SEARCH<x>:TRIGger:A:BUS:  
SOUrce, 2-257
- SEARCH:SEARCH<x>:TRIGger:A:BUS?, 2-240
- SEARCH:SEARCH<x>:TRIGger:A:EDGE:  
SLOpe, 2-258
- SEARCH:SEARCH<x>:TRIGger:A:EDGE:  
SOUrce, 2-258
- SEARCH:SEARCH<x>:TRIGger:A:LEVel, 2-259
- SEARCH:SEARCH<x>:TRIGger:A:LEVel:  
CH<x>, 2-259
- SEARCH:SEARCH<x>:TRIGger:A:LEVel:  
MATH, 2-259
- SEARCH:SEARCH<x>:TRIGger:A:LEVel:  
REF<x>, 2-260
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:  
FUNction, 2-260
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:  
CH<x>, 2-260
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:  
CLOCK:EDGE, 2-261
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:  
CLOCK:SOUrce, 2-261
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:  
D<x>, 2-262
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:  
MATH, 2-262
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:  
REF<x>, 2-263
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:  
INPut:CH<x>, 2-263
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:  
INPut:D<x>, 2-263
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:  
INPut:MATH, 2-264
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:  
INPut:REF<x>, 2-264
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:  
WHEn, 2-265
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:  
WHEn:LESSLimit, 2-265
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:PATtern:  
WHEn:MORELimit, 2-266
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:  
THReshold:CH<x>, 2-266
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:  
THReshold:MATH, 2-266
- SEARCH:SEARCH<x>:TRIGger:A:LOGIc:  
THReshold:REF<x>, 2-267
- SEARCH:SEARCH<x>:TRIGger:A:  
LOWerthreshold:CH<x>, 2-267
- SEARCH:SEARCH<x>:TRIGger:A:  
LOWerthreshold:MATH, 2-267
- SEARCH:SEARCH<x>:TRIGger:A:  
LOWerthreshold:REF<x>, 2-268
- SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:  
POLarity, 2-268
- SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:  
SOUrce, 2-269
- SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:  
WHEn, 2-269
- SEARCH:SEARCH<x>:TRIGger:A:PULSEWidth:  
WIDth, 2-270
- SEARCH:SEARCH<x>:TRIGger:A:RUNT:  
POLarity, 2-270
- SEARCH:SEARCH<x>:TRIGger:A:RUNT:  
SOUrce, 2-270
- SEARCH:SEARCH<x>:TRIGger:A:RUNT:  
WHEn, 2-271
- SEARCH:SEARCH<x>:TRIGger:A:RUNT:  
WIDth, 2-271

- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
CLOCK:EDGE, 2-272
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
CLOCK:SOUrce, 2-272
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
CLOCK:THReshold, 2-273
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:  
SOUrce, 2-273
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATA:  
THReshold, 2-274
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
HOLDTime, 2-274
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
SETTime, 2-274
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
THReshold:CH<x>, 2-275
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
THReshold:REF<x>, 2-275
- SEARCH:SEARCH<x>:TRIGger:A:SETHold:  
THReshold{:MATH|:MATH1}, 2-275
- SEARCH:SEARCH<x>:TRIGger:A:TYPe, 2-277
- SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:  
CH<x>, 2-278
- SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:  
MATH, 2-278
- SEARCH:SEARCH<x>:TRIGger:A:UPPerthreshold:  
REF<x>, 2-279
- SEARCH:SEARCH<x>:TRIGger:A{:TRANsition|:  
RISEFall}:DELtatime, 2-276
- SEARCH:SEARCH<x>:TRIGger:A{:TRANsition|:  
RISEFall}:POLarity, 2-276
- SEARCH:SEARCH<x>:TRIGger:A{:TRANsition|:  
RISEFall}:SOUrce, 2-277
- SEARCH:SEARCH<x>:TRIGger:A{:TRANsition|:  
RISEFall}:WHEn, 2-277
- SEARCH?, 2-236
- SELEct, 2-279
- SELEct:BUS<x>, 2-280
- SELEct:CH<x>, 2-280
- SELEct:CONTRol, 2-281
- SELEct:D<x>, 2-281
- SELEct:MATH[1], 2-282
- SELEct:REF<x>, 2-282
- SET?, 2-283
- SETUP<x>:DATE?, 2-284
- SETUP<x>:LABEL, 2-284
- SETUP<x>:TIME?, 2-285
- \*SRE, 2-285
- Status and Error Command Group, 2-37
- \*STB?, 2-286
- ## T
- TEKSecure, 2-286
- TIME, 2-287
- TOTALuptime?, 2-288
- \*TRG, 2-288
- Trigger Command Group, 2-38
- TRIGger, 2-288
- TRIGger:A:BUS:B<x>:CAN:CONDition, 2-291
- TRIGger:A:BUS:B<x>:CAN:DATA:DIRection, 2-292
- TRIGger:A:BUS:B<x>:CAN:DATA:QUALifier, 2-292
- TRIGger:A:BUS:B<x>:CAN:DATA:SIZE, 2-293
- TRIGger:A:BUS:B<x>:CAN:DATA:VALue, 2-293
- TRIGger:A:BUS:B<x>:CAN:FRAMeType, 2-294
- TRIGger:A:BUS:B<x>:CAN{:IDentifier|:ADDRes}:  
MODE, 2-294
- TRIGger:A:BUS:B<x>:CAN{:IDentifier|:ADDRes}:  
VALue, 2-295
- TRIGger:A:BUS:B<x>:I2C:ADDRes:MODE, 2-295
- TRIGger:A:BUS:B<x>:I2C:ADDRes:TYPe, 2-296
- TRIGger:A:BUS:B<x>:I2C:ADDRes:VALue, 2-296
- TRIGger:A:BUS:B<x>:I2C:CONDition, 2-297
- TRIGger:A:BUS:B<x>:I2C:DATA:DIRection, 2-298
- TRIGger:A:BUS:B<x>:I2C:DATA:SIZE, 2-298
- TRIGger:A:BUS:B<x>:I2C:DATA:VALue, 2-299
- TRIGger:A:BUS:B<x>:LIN:CONDition, 2-299
- TRIGger:A:BUS:B<x>:LIN:DATA:HIVALue, 2-300
- TRIGger:A:BUS:B<x>:LIN:DATA:QUALifier, 2-300
- TRIGger:A:BUS:B<x>:LIN:DATA:SIZE, 2-301
- TRIGger:A:BUS:B<x>:LIN:DATA:VALue, 2-301
- TRIGger:A:BUS:B<x>:LIN:ERRTYPe, 2-302
- TRIGger:A:BUS:B<x>:LIN:IDentifier:VALue, 2-303
- TRIGger:A:BUS:B<x>:PARAllel:VALue, 2-303
- TRIGger:A:BUS:B<x>:RS232C:CONDition, 2-303
- TRIGger:A:BUS:B<x>:RS232C:RX:DATA:  
SIZE, 2-304
- TRIGger:A:BUS:B<x>:RS232C:RX:DATA:  
VALue, 2-304
- TRIGger:A:BUS:B<x>:RS232C:TX:DATA:  
SIZE, 2-305
- TRIGger:A:BUS:B<x>:RS232C:TX:DATA:  
VALue, 2-305
- TRIGger:A:BUS:B<x>:SPI:CONDition, 2-306
- TRIGger:A:BUS:B<x>:SPI:DATA:SIZE, 2-307

- TRIGger:A:BUS:B<x>:SPI:DATA{:IN|:MISO}: VALue, 2-306
- TRIGger:A:BUS:B<x>:SPI:DATA{:OUT|:MOSI}: VALue, 2-307
- TRIGger:A:BUS:SOURce, 2-307
- TRIGger:A:EDGE:COUPling, 2-308
- TRIGger:A:EDGE:SLOPe, 2-309
- TRIGger:A:EDGE:SOURce, 2-309
- TRIGger:A:HOLDoff:TIME, 2-310
- TRIGger:A:LEVel:AUXin, 2-311
- TRIGger:A:LEVel:CH<x>, 2-312
- TRIGger:A:LEVel:D<x>, 2-312
- TRIGger:A:LOGIc:CLASs, 2-313
- TRIGger:A:LOGIc:FUNcTION, 2-314
- TRIGger:A:LOGIc:INPut:CH<x>, 2-315
- TRIGger:A:LOGIc:INPut:CLOCK:EDGE, 2-315
- TRIGger:A:LOGIc:INPut:CLOCK:SOURce, 2-316
- TRIGger:A:LOGIc:INPut:D<x>, 2-316
- TRIGger:A:LOGIc:INPut?, 2-315
- TRIGger:A:LOGIc:PATtern:DELtAtime, 2-317
- TRIGger:A:LOGIc:PATtern:INPut:D<x>, 2-317
- TRIGger:A:LOGIc:PATtern:WHEn, 2-318
- TRIGger:A:LOGIc:PATtern:WHEn: LESSLimit, 2-319
- TRIGger:A:LOGIc:PATtern:WHEn: MORELimit, 2-319
- TRIGger:A:LOGIc:PATtern?, 2-317
- TRIGger:A:LOGIc:THREshold: (CH<x>|D<x>), 2-320
- TRIGger:A:LOWerthreshold:CH<x>, 2-320
- TRIGger:A:LOWerthreshold{:EXT|:AUX}, 2-321
- TRIGger:A:PULSe:CLASs, 2-322
- TRIGger:A:PULSEWidth:POLArity, 2-323
- TRIGger:A:PULSEWidth:SOURce, 2-323
- TRIGger:A:PULSEWidth:WHEn, 2-324
- TRIGger:A:PULSEWidth:WIDth, 2-325
- TRIGger:A:PULSEWIDth?, 2-323
- TRIGger:A:RUNT:POLArity, 2-325
- TRIGger:A:RUNT:SOURce, 2-326
- TRIGger:A:RUNT:WIDth, 2-327
- TRIGger:A:SETHold:CLOCK:EDGE, 2-328
- TRIGger:A:SETHold:CLOCK:SOURce, 2-329
- TRIGger:A:SETHold:CLOCK:THREshold, 2-329
- TRIGger:A:SETHold:CLOCK?, 2-328
- TRIGger:A:SETHold:DATA:SOURce, 2-330
- TRIGger:A:SETHold:DATA:THREshold, 2-331
- TRIGger:A:SETHold:DATA?, 2-330
- TRIGger:A:SETHold:HOLDTime, 2-332
- TRIGger:A:SETHold:SETTime, 2-332
- TRIGger:A:SETHold:THREshold:CH<x>, 2-332
- TRIGger:A:SETHold:THREshold:D<x>, 2-333
- TRIGger:A:UPPerthreshold:CH<x>, 2-337
- TRIGger:A:VIDEo:HOLDoff:FIELD, 2-338
- TRIGger:A:VIDEo:LINE, 2-338
- TRIGger:A:VIDEo:POLArity, 2-339
- TRIGger:A:VIDEo:SOURce, 2-340
- TRIGger:A:VIDEo:STANdard, 2-340
- TRIGger:A:VIDEo{:SYNC|:FIELD}, 2-341
- TRIGger:A{:TRANsition|:RISEFall}: DELtAtime, 2-334
- TRIGger:A{:TRANsition|:RISEFall}: POLArity, 2-334
- TRIGger:A{:TRANsition|:RISEFall}: SOURce, 2-335
- TRIGger:A{:TRANsition|:RISEFall}: WHEn, 2-335
- TRIGger:A{:TRANsition|:RISEFall}?, 2-333
- TRIGger:EXTErnal:PROBe, 2-342
- TRIGger:EXTErnal:YUNIts?, 2-342
- TRIGger:A, 2-289
- TRIGger:A:BUS, 2-290
- TRIGger:A:EDGE?, 2-308
- TRIGger:A:HOLDoff?, 2-310
- TRIGger:A:LEVel, 2-311
- TRIGger:A:LOGIc?, 2-313
- TRIGger:A:MODE, 2-321
- TRIGger:A:PULSe?, 2-322
- TRIGger:A:RUNT?, 2-325
- TRIGger:A:RUNT:WHEn, 2-326
- TRIGger:A:SETHold?, 2-328
- TRIGger:A:TYPE, 2-336
- TRIGger:A:VIDEo?, 2-337
- TRIGger:EXTErnal?, 2-341
- TRIGger:FREQuency?, 2-289
- TRIGger:STATE?, 2-342
- \*TST?, 2-343

## U

- UNLock, 2-343
- USBDevice:CONFigure, 2-344
- USBTMC, 2-344
- USBTMC:PRODUCTID:DECimal?, 2-344
- USBTMC:PRODUCTID:HEXadecimal?, 2-345
- USBTMC:SERIALnumber?, 2-345
- USBTMC:VENDORID:DECimal?, 2-345
- USBTMC:VENDORID:HEXadecimal?, 2-345

**V**

VERBose, 2-346  
Vertical Command Group, 2-45

**W**

\*WAI, 2-347  
Waveform Transfer Command Group, 2-48  
WAVFrm?, 2-347  
WFMInpre:COMPosition, 2-350  
WFMInpre?, 2-348  
WFMInpre:BIT\_Nr, 2-348  
WFMInpre:BN\_Fmt, 2-349  
WFMInpre:BYT\_Nr, 2-349  
WFMInpre:BYT\_Or, 2-350  
WFMInpre:ENCdg, 2-351  
WFMInpre:FILTERFreq, 2-351  
WFMInpre:NR\_Pt, 2-352  
WFMInpre:PT\_Fmt, 2-352  
WFMInpre:PT\_Off, 2-353  
WFMInpre:XINcr, 2-353  
WFMInpre:XUNit, 2-354  
WFMInpre:XZEro, 2-355  
WFMInpre:YMUlt, 2-355  
WFMInpre:YOFF, 2-356  
WFMInpre:YUNit, 2-357  
WFMInpre:YZEro, 2-357  
WFMOutpre:COMPosition, 2-361  
WFMOutpre:FILTERFreq?, 2-361  
WFMOutpre:RECOrdlength?, 2-362  
WFMOutpre?, 2-358

WFMOutpre:BIT\_Nr, 2-358  
WFMOutpre:BN\_Fmt, 2-359  
WFMOutpre:BYT\_Nr, 2-360  
WFMOutpre:BYT\_Or, 2-360  
WFMOutpre:ENCdg, 2-362  
WFMOutpre:NR\_Pt?, 2-363  
WFMOutpre:PT\_Fmt?, 2-363  
WFMOutpre:PT\_Off?, 2-363  
WFMOutpre:PT\_ORDER?, 2-364  
WFMOutpre:WFIId?, 2-364  
WFMOutpre:XINcr?, 2-365  
WFMOutpre:XUNit?, 2-366  
WFMOutpre:XZEro?, 2-366  
WFMOutpre:YMUlt?, 2-367  
WFMOutpre:YOFF?, 2-367  
WFMOutpre:YUNit?, 2-368  
WFMOutpre:YZEro?, 2-368

**Z**

Zoom Command Group, 2-55  
ZOOM:ZOOM1:HORizontal:POSITION, 2-370  
ZOOM:ZOOM1:HORizontal:SCALE, 2-370  
ZOOM?, 2-369  
ZOOM{:MODE|:STATE}, 2-369  
ZOOM:ZOOM1?, 2-369  
ZOOM:ZOOM1:FACtor?, 2-370  
ZOOM:ZOOM1:POSITION, 2-371  
ZOOM:ZOOM1:SCALE, 2-371  
ZOOM:ZOOM1:STATE, 2-372