Tektronix®

MSO/DPO5000/B, DPO7000/C, DPO70000/B/C/D/DX/SX, DSA70000/B/C/D, and MSO70000/C/DX Series Digital Oscilloscopes

Programmer Manual

077-0010-23

Tektronix®

MSO/DPO5000/B, DPO7000/C, DPO70000/B/C/D/DX/SX, DSA70000/B/C/D, and MSO70000/C/DX Series Digital Oscilloscopes

Programmer Manual

www.tek.com

077-0010-23

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.

UltraSync, FastFrame, OpenChoice, iView, Pinpoint, RT-Eye, MyScope, TekLink, TekVPI, MultiView Zoom, and DPX are trademarks of Tektronix, Inc.

DPO7000C, MSO/DPO5000/B, DPO7000, DPO70000/B/C/D/DX/SX, DSA70000/B/C/D, and MSO70000/C/DX Series Programmer Online Help, 076-0021-22

Contacting Tektronix

Tektronix, Inc.
14150 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.tek.com to find contacts in your area.

Table of Contents

Preface iii
Getting started 1-1
Setting up remote communications 1-2
Command syntax 2-1
Backus-Naur Form notation 2-1
Command and query structure 2-1
Clearing the instrument 2-3
Command entry 2-3
Constructed mnemonics 2-5
Argument types 2-6
Command groups 2-9
Acquisition command group 2-9
Alias command group 2-11
Bus command group 2-12
Calibration command group 2-16
Cursor command group 2-17
Diagnostics command group 2-19
Digital command group 2-20
Display control command group 2-21
E-mail command group 2-23
Error Detector command group 2-25
File system command group 2-34
Hard copy command group 2-35
Histogram command group 2-36
Horizontal command group 2-37
Limit Test command group 2-40
Link Training command group 2-41
Low Speed Serial Trigger command group 2-42
Mask command group 2-48
Math command group 2-52
Measurement command group 2-54
Miscellaneous command group 2-57
Save and Recall command group 2-59
Save On command Group 2-60
Search and Mark command group 2-61
Status and Error command group 2-79
Trigger command group 2-80
Vertical command group 2-94
Waveform Transfer command group 2-98

Zoom command group 2-103
Commands listed in alphabetical order..... 2-105
Status and events 3-1
Registers 3-1
*PSC command 3-4
Queues 3-4
Event handling sequence..... 3-5
Synchronization methods 3-6
Appendix A: Character set..... A-1
Appendix B: Reserved words B-1
Appendix C: Factory default setup values C-1
Default setup C-1
Appendix D: GPIB interface specifications..... D-1
Interface messages..... D-1
GPIB functions..... D-1
Glossary
Index

Preface

This programmer guide provides you with the information required to use Programmable Interface commands for remotely controlling your instrument.

Documentation

Quick Start User Manual. The user manual has information about installing and operating the instrument. It also provides concepts and theories about using the instrument that are not covered in the online help.

Online Help. This is an online help system that is integrated with the User Interface application that ships with this product. The online help provides in-depth operation and user interface help.

Getting Started with OpenChoice ™ Solutions Manual. A book that explores some options for getting data from your instrument into any one of several available analysis tools.

Specifications and Performance Verification. Instrument specifications and a performance verification procedure.

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

Other Included Documentation. Installation booklets are included in the Product Software and Operating System Restore Software packages.

Service Manual. The service manual includes procedures to service the instrument to the module level.

The programmer manual is divided into the following major topics:

- **Getting started.** This topic introduces you to the programming information and provides basic information about setting up your instrument for remote control.
- **Command groups.** This topic contains all the commands listed in functional groups. Each group consists of an overview of the commands in that group and a table that lists all the commands and queries for that group. You can click a command in the listing to display a detailed description of the command.
- **Command syntax.** This topic provides an overview of the command syntax that you use to communicate with the instrument and other general information about commands, such as how commands and queries are constructed, how to enter commands, constructed mnemonics, and argument types.
- **Status and events.** This topic discusses the status and event reporting system for the GPIB interfaces. This system informs you of certain significant events that occur within the instrument. Topics that are discussed include registers, queues, event handling sequences, synchronization methods, and messages that the instrument might return, including error messages.
- **Miscellaneous.** This topic contains miscellaneous information, such as a list of reserved words, a table of the factory initialization (default) settings, and interface specifications that can be helpful when using commands to remotely control the instrument.

New in the programmer manual

The following major changes were made to this version of the programmer manual:

- Added the following new or changed commands or groups of commands:
  - BUS:B<x>:PCIE:BITRate
  - SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CONDition
  - SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:LONGORDERedset
  - CH<x>:INVert
  - LINKTRaining:EQUalization
  - LINKTRaining:ACQTime
  - LINKTRaining:SETUP
  - CH<x>:OPTIcal:RCVR
  - CH<x>:OPTIcal:WLENgth
  - CH<x>:OPTIcal:WLENgth:LIST?

Getting started

This programmer guide provides you with the information required to use Programmable Interface commands for remotely controlling your instrument. With this information, you can write computer programs that will perform functions such as setting the front panel controls, taking measurements, performing statistical calculations, and exporting data for use in other programs, such as spreadsheets.

NOTE. *This programmer guide covers many oscilloscope models from Tektronix. Most of the command listed here are available for all of these covered models. However, some commands are available only on certain models; these commands are identified by a NOTE in the description.*

In addition to the traditional GPIB electronic interface (referred to as the physical GPIB interface), your instrument is provided with a *TekVISA* GPIB-compatible interface (referred to as the virtual GPIB interface). This is a software Application Programming Interface (API) which enables you to communicate with the instrument in a variety of ways, including via the Internet. With the following two exceptions, these interfaces are completely independent:

- HEADER. Command headers enabled or disabled on one interface are correspondingly enabled or disabled on the other interface. Refer to the command descriptions for more detailed information.
- VERBOSE. Verbosity enabled or disabled on one interface is correspondingly enabled or disabled on the other interface. Refer to the command description for more detailed information.

Most examples in this document require that both `HEADER` and `VERBOSE` are `ON`.

Refer to *Documentation* in the Preface for information on related manuals and documents.

Setting up remote communications

On MSO/DPO5000/B Series instruments only: The instruments require a TEK-USB-488 adapter. You can remotely communicate between your oscilloscope and PC via Ethernet, USB, and GPIB.

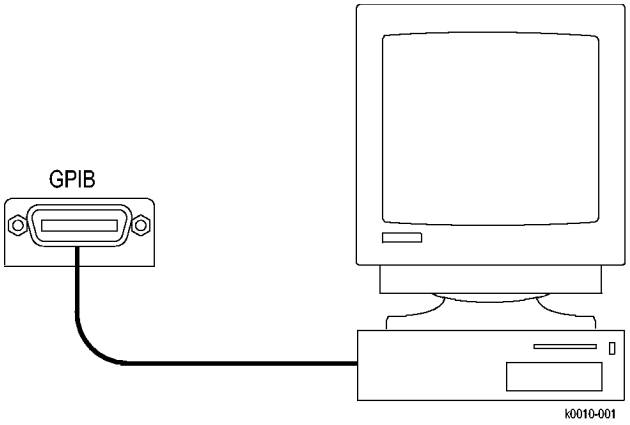
- To use Ethernet, start by connecting an appropriate Ethernet cable to the Ethernet port (RJ-45 connector) on the rear panel of your oscilloscope. This connects the oscilloscope to a 10/100/1000 Base-T local area network.
- To use USB, start by connecting an appropriate USB cable to the USB device port on the rear panel of your oscilloscope. With USB, the system automatically configures itself.
- To use GPIB, start by connecting an appropriate USB cable to the USB 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 or other GPIB-enabled instrument. You can apply power to the TEK-USB-488 adapter in one of the following two ways:
  - With an appropriate power supply connected to wall power and the adapter.
  - By connecting a USB cable to a Host port on the oscilloscope and the device (B type) connector on the adapter. Without a power connection, the adapter will not function properly.

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

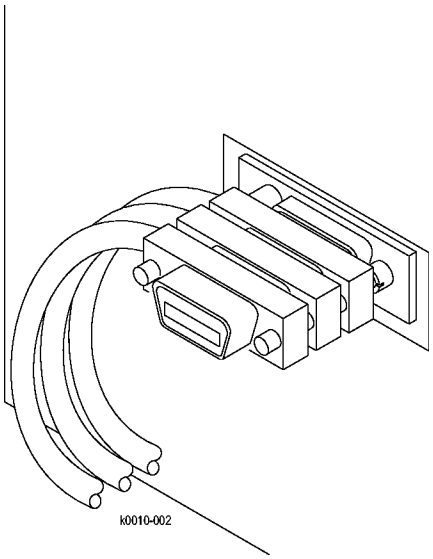
- A unique device address must be assigned to each device on the bus. No two devices can share the same device address.
- No more than 15 devices can be connected to any one line.
- One device should be connected for every 6 feet (2 meters) of cable used.
- No more than 65 feet (20 meters) of cable should 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.

Connecting to the instrument

Your instrument has a 24-pin GPIB connector on its rear (side) panel. (This connector is available on all instruments except MSO/DPO5000/B Series instruments.) This connector has a D-type shell and conforms to IEEE Std 488.1¾1987. Attach an IEEE Std 488.1¾1987 GPIB cable to this connector and to your controller as shown in the following figure.



If necessary, the GPIB connectors can be stacked as shown in the following figure:



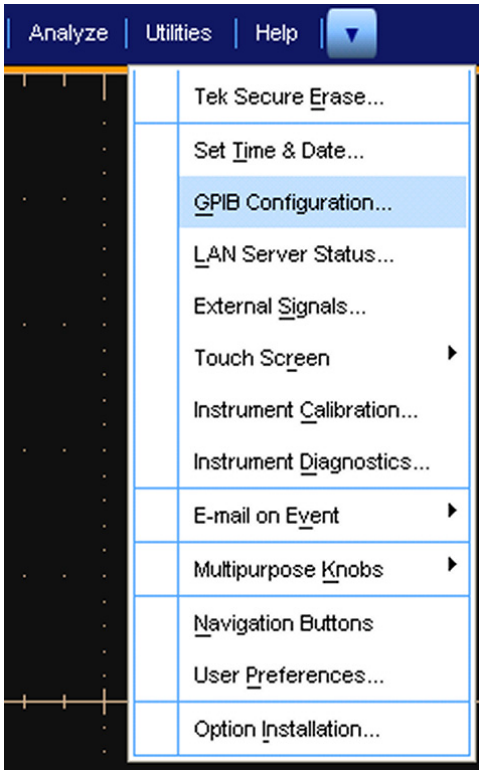
Setting the GPIB address

To function correctly, your instrument must have a unique device address. The default settings for the GPIB configuration are:

- GPIB Address 1.
- GPIB Mode Talk/Listen

To change either of the GPIB settings, do the following:

1. Select GPIB Configuration from the Utilities menu.



2. Click the Configuration Talk/Listen button.

The following screen appears on MSO/DPO5000/B Series instruments:

3. Change the GPIB Address to a unique address.
4. Click the Close button. The instrument is now set up for bidirectional communication with your controller.

Command syntax

You can control the operations and functions of the instrument through the use of commands and queries. The following related topics describe the syntax of these commands and queries, as well as the conventions that the instrument 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 guide 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 instrument settings or tell the instrument to perform a specific action. Queries cause the instrument 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 on 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 instrument must have in order to execute the command or query. Command messages can contain five element types, defined in the following table.

Table 2-2: Command message elements

Symbol	Meaning
<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 an asterisk (*).
<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 instrument 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 can affect those at a lower level. The leading colon (:) always returns you to the base of the command tree.

Queries

Queries cause the instrument 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 in the command tree unless otherwise noted. These branch queries return information about all the mnemonics below the specified branch or level. For example, HIStogram:STATistics:STDdev? returns the standard deviation of the histogram, HIStogram:STATistics? returns all the histogram statistics, and HIStogram? returns all the histogram parameters.

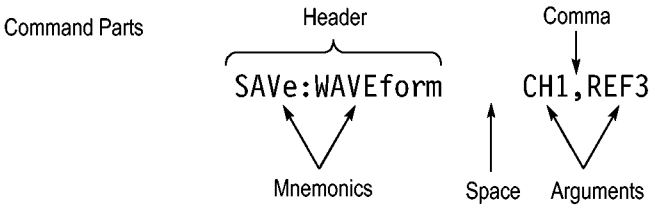
Headers

You can use the HEADer command to control whether the instrument returns headers as part of the query response. If header is on, the query returns command

headers and then formats itself as a valid set command. If header is off, the response includes only the values. This can make it easier to parse and extract information from the response. The table 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 instrument

You can use the selected Device Clear (DCL) GPIB function to clear the Output Queue and reset the instrument to accept a new command or query. Refer to your GPIB library documentation for details about the Device Clear operation.

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 instrument ignores commands consisting of any combination of white space characters and line feeds.

Abbreviating

You can abbreviate many instrument commands. Commands in this document show the abbreviations in capitals. For example, you can enter the command ACQuire:NUMAvg simply as ACQ:NUMA or acq:numa. Abbreviation rules can change over time as new instrument 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 instrument executes concatenated commands in the order received. When concatenating, 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 10`, can be concatenated into the following single command:

   ```
   TRIGger:MODe NORMal;:ACQuire:NUMAVg 10
   ```

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 ENVelope` and `ACQuire:NUMAVg 10` into a single command:

   ```
   ACQuire:MODe ENVelope; NUMAVg 10
   ```

   The longer version works equally well:

   ```
   ACQuire:MODe ENVelope;:ACQuire:NUMAVg 10
   ```

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

   ```
   ACQuire:MODe ENVelope;*OPC
   ```

   Any commands that follow will be processed as if the star command was not there, so the commands `ACQuire:MODe ENVelope;*OPC;NUMAVg 10` will set the acquisition mode to envelope and set the number of acquisitions for averaging to 10.

4. When you concatenate queries, the responses to all the queries are concatenated into a single response message. For example, if the display imageview color is temperature and the display recordview color is spectral, the concatenated query `DISplay:COLOr:PALETTE:IMAGEVIEW?; RECORDVIEW?` will return the following.

   If the header is on:

   ```
   :DISPLAY:COLOR:PALETTE:IMAGEVIEW TEMPERATURE;
   :DISPLAY:COLOR:PALETTE:RECORDVIEW SPECTRAL
   ```

   If the header is off:

   ```
   TEMPERATURE;SPECTRAL
   ```

5. Set commands and queries can 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:

`DISplay:PERSistance:RESET;ACQuire:NUMAVg 10` (no colon before ACQuire)

`DISplay:GRAticule FULl;:FILTer SINX` (extra colon before `FILTer;` use `DISplay:GRAticule FULl;FILTer SINX` instead)

`DISplay:PERSistance:RESET;:*OPC` (colon before a star (*) command)

`DISplay:COLOr:MATHCOLOr DEFAULT;COLOr:REFCOLOr INHERIT` (levels of the mnemonics are different; either remove the second use of `COLOr` or place `:DISPlay:` in front of `COLOr:REFCOLOr INHERIT`)

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 linefeed (LF) character.

This instrument does not support ASCII LF only message termination. The instrument 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: 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 2-6: Math specifier mnemonics

Symbol	Meaning
Math<x>	A math waveform specifier; <x> is 1 through 4.

Measurement specifier mnemonics

Commands can specify which measurement to set or query as a mnemonic in the header. Up to eight automated measurements can be displayed.

Table 2-7: Measurement specifier mnemonics

Symbol	Meaning
MEAS<x>	A measurement specifier; <x> is 1 through 8.

Channel mnemonics

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

Table 2-8: Channel Mnemonics

Symbol	Meaning
CH<x>	A channel specifier; <x> is 1 through 4.
SOURCE<1..2>	A source specifier, the source can be 1 through 2.

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> is 1 thru 4.

Argument types

Numeric

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

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

Most numeric arguments will be automatically forced to a valid setting, either by 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 if 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, but is treated as just another character in the string.
7. The maximum length of a quoted string returned from a query is 255 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

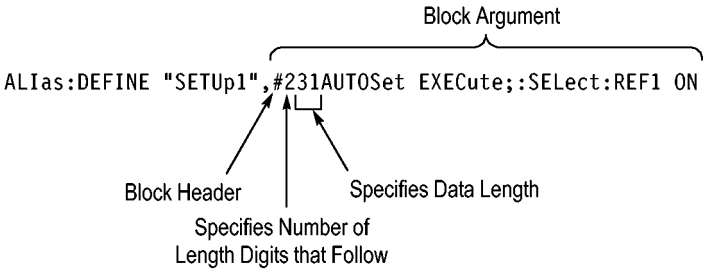
Several instrument commands use a block argument form (see the following table).

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.

NOTE. *The digit <NZDig> is in hexadecimal format. This deviates slightly from the IEEE 488.2 specification that it be in decimal format, as extra allowances must be made for data lengths that are greater than 999,999,999 <DChar> elements (for example 500M record lengths at 2 bytes per point).*



Command groups

The programmable interface conforms to Tektronix standard codes and formats except where noted. The GPIB interface also conforms to IEEE Std 488.2-1987 except where noted.

Acquisition command group

Acquisition commands set up the modes and functions that control how the instrument acquires signals and processes them into waveforms. Using these commands for acquiring waveforms, you can do the following:

- Start and stop acquisitions.
- Control whether each waveform is simply acquired, averaged, or enveloped over successive acquisitions of that waveform.
- Set the controls or conditions that start and stop acquisitions.
- Determine the action the system takes upon completing an acquisition, such as saving all waveforms and taking a measurement.
- Control acquisition of acquired channel waveforms.
- Set acquisition parameters.

Table 2-13: Acquisition commands

Command	Description
ACQuire:ENHANCEDEnob	Sets or queries the state of the enhanced effective number of bits.
ACQuire:ENHANCEDEnob:STATE?	Returns the state of the enhanced effective number of bits.
ACQuire:INTERPEightbit	Sets or queries the interpolation acquisition mode
ACQuire:MAGnivu	Sets or queries the MagniVu feature ***NOTE.*** *Not available on some models*
ACQuire:MODe	Sets or queries acquisition mode
ACQuire:MODe:ACTUal?	Return the actual acquisition mode that the hardware used for HiRes and PkDetect mode
ACQuire:NUMFRAMESACQuired?	Returns the number of acquisitions that have occurred
ACQuire:NUMACq?	Returns the number of waveform acquisitions that have occurred since starting acquisition with the `ACQuire:STATE RUN` command
ACQuire:NUMAVg	Sets or queries number of acquisitions for an averaged waveform
ACQuire:NUMEnv	Sets or queries number of acquisitions for envelope waveform
ACQuire:NUMSAMples	Sets or queries the number of samples that make up a WfmDB for single sequence mode and Mask Pass/Fail Completion Test
ACQuire:SAMPlingmode	Sets or queries the sampling mode
ACQuire:STATE	Starts, stops, or returns acquisition state

Table 2-13: Acquisition commands (cont.)

Command	Description
ACQuire:STOPAfter	Sets or queries whether the acquisition is continuous or single sequence
FASTAcq?	Enables, disables, or returns state of Fast Acquisition mode
FASTAcq:HIACQRATE	Sets or queries the state of FastAcq optimization
FASTAcq:STATE	Returns the Fast Acquisition state

Alias command group

Alias commands allow you to define new commands as a sequence of standard commands. You might 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 IEEE 488.2 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 program date, 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.
- The Alias commands are defined in Tektronix Standard Codes and Formats. Deviations between that standard and what is specified here will be considered errors unless specifically noted in the command description in this document.

Table 2-14: Alias commands

Command	Description
ALIas	Sets or queries the alias state
ALIas:CATalog?	Returns a list of the currently defined alias labels
ALIas:DEFine	Assigns a sequence of program messages to an alias label
ALIas:DELEte	Removes a specified alias
ALIas:DELEte:ALL	Deletes all existing aliases
ALIas:DELEte:NAMe	Removes a specified alias
ALIas:STATE	Sets or queries the alias state

Bus command group

NOTE. *Not available on some models.*

Use the commands in the Bus Command Group to configure a bus. These commands let you:

- Specify the bus type
- Specify the digital signals to be used in the bus
- Specify its display style

Bus Mnemonics

Commands specify the bus to use as a mnemonic in the header.

NOTE. *For MIPI general commands, currently only Lane 1 is supported.*

Table 2-15: Bus mnemonics

Symbol	Meaning
B<x>	A bus specifier; <x> is 1 through 16

Table 2-16: Bus commands

Command	Description
BUS:B<x>:CAN:SOUrce	Sets or queries the CAN source channel
BUS:B<x>:CAN:BITRate	Sets or queries the CAN bitrate
BUS:B<x>:CAN:BITRate:VALue	Sets or queries CAN bitrate value
BUS:B<x>:CAN:PRObe	Sets or queries CAN probe type
BUS:B<x>:DISplay:DECOde:FILe	Sets or queries the symbol table file for the specified bus
BUS:B<x>:DISplay:DECOde:STAte	Sets or queries whether the specified bus is enabled to display symbolic decode of its busform values
BUS:B1<x>:DISplay:HIERarchical	This command sets or queries the display of a bus layer on or off.
BUS:B1<x>:DISplay:LAYout	This command sets or queries the format a bus layer should use.
BUS:B<x>:ETHERnet:PRObe	This command specifies the Ethernet probe type: differential or single-ended.
BUS:B<x>:ETHERnet:SOUrce	Specifies the Ethernet data source for differential input
BUS:B<x>:ETHERnet:SOUrce:DMINus	This command specifies the Ethernet data source for D- input for differential probing.
BUS:B<x>:ETHERnet:SOUrce:DPLUs	This command specifies the Ethernet data source for the D+ input for differential probing.
BUS:B<x>:ETHERnet:TYPe	Specifies the Ethernet standard type: 10Base-T or 100Base-T
BUS:B<x>:FLEXRAY:BITRate	Sets or queries the FLEXRAY bus bit rate

Table 2-16: Bus commands (cont.)

Command	Description
BUS:B<x>:FLEXRAY:BITRate:VALue	Sets or queries the bus is specified by x
BUS:B<x>:FLEXRAY:CHANnel	Sets or queries the FLEXRAY bus channel
BUS:B<x>:FLEXRAY:SOUrce	Sets or queries the FLEXRAY bus source
BUS:B<x>:FLEXRAY{:PROBe|:SIGnal}	Sets or queries the FLEXRAY probe
BUS:B<x>:I2C:CLOCk:SOUrce	Sets or queries the I2C clock (SCLK) source for the specified bus
BUS:B<x>:I2C:DATa:SOUrce	Sets or queries the I2C data (SDA) source for the specified bus
BUS:B<x>:I2C:RWINADDR	Determines whether decoded I2C slave addresses are pure seven-bit values, or have the R/W* combined with them
BUS:B<x>:LABel	Sets or queries the waveform label for the specified bus
BUS:B<x>:LIN:BITRate	Sets or queries the LIN bus bit rate
BUS:B<x>:LIN:BITRate:VALue	Sets or queries the LIN bus bitrate value
BUS:B<x>:LIN:IDFORmat	Sets or queries LIN bus id format
BUS:B<x>:LIN:POLarity	Sets or queries the LIN bus polarity
BUS:B<x>:LIN:SOUrce	Sets or queries sets the LIN bus source
BUS:B<x>:LIN:STANDard	Sets or queries the LIN bus standard
BUS:B<x>:MIL1553B:POLarity	Sets the MIL-STD-1553 bus polarity to normal or inverted
BUS:B<x>:MIL1553B:RESPonsetime:MAXimum	Specifies the maximum response time to a valid command issued
BUS:B<x>:MIL1553B:RESPonsetime:MINimum	Specifies the minimum response time to a valid command issued
BUS:B<x>:MIL1553B:SOUrce	Sets or queries sets the MIL-STD-1553 bus source
BUS:B<x>:MIPICSITWo:CLOCk:SOUrce	Sets or queries the MIPI CSI2 clock source for the specified bus
BUS:B<x>:MIPICSITWo:CLOCk:TYPe	Sets or queries the MIPI CSI2 clock source type for the specified bus
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DIFFerential	Sets or queries the differential source for the specified lane of the specified MIPI CSI2 bus
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DMINUS	Sets or queries the D Minus source for the specified lane of the specified MIPI CSI2 bus
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DPLUS	Sets or queries the D Plus source for the specified lane of the specified MIPI CSI2 bus
BUS:B<x>:MIPICSITWo:LANE<x>:TYPe	Sets or queries the lane source type for the specified MIPI CSI2 bus
BUS:B<x>:MIPIDSIOne:CLOCk:SOUrce	Sets or queries the clock source for the specified MIPI DSI1 bus
BUS:B<x>:MIPIDSIOne:CLOCk:TYPe	Sets or queries the MIPI DSI1 clock source type for the specified bus
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DIFFerential	Sets or queries the differential source for the specified lane of the specified MIPI DSI1 bus
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DMINUS	Sets or queries the D Minus source for the specified lane of the specified MIPI DSI1 bus
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DPLUS	Sets or queries the D Plus source for the specified lane of the specified MIPI DSI1 bus
BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe	Sets or queries the lane source type for the specified MIPI DSI1 bus
BUS:B<x>:PARallel:CLOCk:EDGE	Determines which edges of its clock signal cause a clocked parallel bus to sample new states

Table 2-16: Bus commands (cont.)

Command	Description
BUS:B<x>:PARallel:CLOCk:SOUrce	Sets or queries the Parallel clock source for the specified bus
BUS:B<x>:PARallel:ISCLOCKED	Determines whether the bus operates in a clocked or asynchronous fashion
BUS:B<x>:PARallel:SOURCES	Sets or queries the member signals for the Parallel mode of the bus to or reports them as an MSB-to-LSB ordered list
BUS:B<x>:PCIE:BITRate	Sets or queries the PCIE bus bit rate
BUS:B<x>:PCIE:BITRate:VALue	Sets or queries the PCIE bus bit rate value
BUS:B<x>:PCIE:HYSTeresis	Sets or queries the PCIE bus hysteresis
BUS:B<x>:PCIE:LANE	Sets or queries the PCIE bus lane
BUS:B<x>:PCIE:SOUrce	Sets or queries the PCIE bus source
BUS:B<x>:POSition	Sets or queries the position waveform for the specified bus
BUS:B<x>:RS232C:BITRate	Sets or queries the RS-232 bit rate for the specified bus
BUS:B<x>:RS232C:DATABits	Sets or queries the number of RS-232 data bits for the specified bus
BUS:B<x>:RS232C:DELIMiter	Sets or queries the RS-232 delimiting value for a packet on the specified bus
BUS:B<x>:RS232C:DISplaymode	Sets or queries the display mode for the specified bus
BUS:B<x>:RS232C:PARity	Sets or queries the RS-232 parity for the specified bus
BUS:B<x>:RS232C:POLarity	Sets or queries the RS-232 polarity for the specified bus
BUS:B<x>:RS232C:SOUrce	Sets or queries the RS-232 polarity for the specified bus
BUS:B<x>:S8B10B:BITRate	Sets the bus data for the specified bus to a standard rate in bits per second, or enables you to specify a custom data rate using the value command
BUS:B<x>:S8B10B:BITRate:VALue	Sets the data rate for the specified bus to a rate that you specify in bits per second
BUS:B<x>:S8B10B:HYSTeresis	Sets or queries the hysteresis for the specified bus
BUS:B<x>:S8B10B:SOUrce	Sets or queries the signal sources for the specified bus
BUS:B<x>:SPI:BITOrder	Sets or queries the shift direction used to de-serialize data for the SPI mode of the bus
BUS:B<x>:SPI:CLOCk:POLarity	Sets or queries the SPI clock (SCLK) polarity for the specified bus
BUS:B<x>:SPI:CLOCk:SOUrce	Sets or queries the SPI clock (SCLK) source for the specified bus
BUS:B<x>:SPI:DATa:POLarity	Sets or queries the SPI data (DATA) polarity for the specified bus
BUS:B<x>:SPI:DATa:SIZe	Sets or queries the number of bits per word for the specified bus
BUS:B<x>:SPI:DATa:SOUrce	Sets or queries the SPI data (DATA) source for the specified bus
BUS:B<x>:SPI:FRAMING	Sets or queries the SPI bus framing
BUS:B<x>:SPI:IDLETime	Sets or queries the SPI bus idle time
BUS:B<x>:SPI:SELect:POLarity	Sets or queries the SPI Slave Select (SS) polarity for the specified bus
BUS:B<x>:SPI:SELect:SOUrce	Sets or queries the SPI Slave Select (SS) source for the specified bus
BUS:B<x>:TYPe	Sets or queries the bus type specified
BUS:B<x>:USB:BITRate	Sets or queries the USB bit rate for the specified bus
BUS:B1<x>:USB:HYSTeresis	This command sets or queries the hysteresis for USB Super Speed.

Table 2-16: Bus commands (cont.)

Command	Description
BUS:B<x>:USB:PRObe	Sets or queries the type of probe connected to the USB signal for the specified bus
BUS:B<x>:USB:SOUrce	Sets or queries the USB Data Source for the specified bus
BUS:B<x>:USB:SOUrce:DMINus	Sets or queries the USB Data Source for D- input for the specified bus
BUS:B<x>:USB:SOUrce:DPLUs	Sets or queries the USB Data Source for D+ input for the specified bus
BUS:CH<x>:LOWTHRESHold	Sets or queries the low threshold value for the analog source in the bus (USB differential)
BUS:CH<x>:THRESHold	If there is a high and low threshold for the analog source in the bus, this command sets or queries the high threshold value. Otherwise, this command sets or queries the threshold value
BUS:MATH<x>:LOWTHRESHold	Sets or queries the low threshold value of the mathematical waveform for the bus (USB differential)
BUS:MATH<x>:THRESHold	If there is a high and low threshold for the mathematical waveform in the bus, this command sets or queries the high threshold value. Otherwise, this command sets or queries the threshold value
BUS:REF<x>:THRESHold	If there is a high and low threshold for the reference waveform in the bus, this command sets or queries the high threshold value. Otherwise, this command sets or queries the threshold value
SELect:B<x>	Sets or queries the display state for the specified bus

Calibration command group

The Calibration commands provide information about the current state of instrument calibration and allow you to initiate internal signal path calibration (SPC).

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes. Any remote command that performs an action on the oscilloscope is also disabled until the SPC is complete.*

Table 2-17: Calibration commands

Command	Description
CALibrate?	Returns the internal and factory calibration status
*CAL?	Instructs the instrument to perform self-calibration and returns the calibration status when complete. Takes several minutes to run
CALibrate:CALProbe:CH<x>?	Performs a probe calibration for the selected channel and returns the calibration status
CALibrate:INTERNal	Starts the internal signal path calibration. Takes several minutes to run
CALibrate:INTERNal:STARt	Starts the internal signal path calibration
CALibrate:INTERNal:STATus?	Returns the status of the internal signal path calibration
CALibrate:PRObestate:CH<x>?	Returns the probe calibration status for the probe of the selected channel
CALibrate:RESults?	Returns the status of all calibration subsystems without performing an SPC operation
CALibrate:RESults:SPC?	Returns the results of the last SPC operation

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 waveform source, cursor position, and cursor color.

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

- **Off.** Shuts off the display of all cursors.
- **Vertical bars.** Displays vertical bar cursors, which provide traditional horizontal unit readouts for Cursor 1 (bar1), Cursor 2 (bar2), the delta between them, and 1/delta (results in frequency when the horizontal unit is time).
- **Horizontal bars.** Displays horizontal bar cursors, which provide traditional vertical unit readouts for Cursor 1 (bar1), Cursor 2 (bar2), and the delta between them.
- **Waveform cursors.** Consists of two cursors you can independently assign to a waveform. These cursors provide the same readouts that the vertical and horizontal bar cursors provide. Waveform cursors enable you to conveniently measure waveform amplitude and time. In XY or XYZ format, waveform cursors indicate the amplitude position of an XY pair (Ch1 vs Ch2 voltage, where Ch1 is the X axis and Ch2 is the Y axis) relative to the trigger.
- **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. Screen cursors, depending on the style selected, consist of the intersection of a vertical and horizontal line, an X, or a vertical line with an X. These cursors have no association with any waveform, except that they inherit the color of the waveform they are assigned to.

Table 2-18: Cursor commands

Command	Description
CURSor?	Returns all cursor settings
CURSor:FUNCtion	Sets or queries the cursor type
CURSor:HBArs?	Returns hbar cursor settings
CURSor:HBArs:DELTa?	Returns hbars cursors vertical difference
CURSor:HBArs:POSITION<x>	Sets or queries the hbar cursor<x> vertical position
CURSor:HBArs:UNIts?	Returns hbar cursor units
CURSor:LINESTyle	Sets or queries the cursor line style
CURSor:MODe	Sets or queries whether cursors move in unison or separately
CURSor:SCREEN:STYle	Sets or queries the cursor type for screen mode
CURSor:SCREEN:XPOSITION<x>	Sets or queries the x position of the specified screen cursor
CURSor:SCREEN:YPOSITION<x>	Sets or queries the y position of the specified screen cursor
CURSor:SOUrce<x>	Sets or queries the source for cursor <x>
CURSor:STATE	Turns cursors on or off or returns their state

Table 2-18: Cursor commands (cont.)

Command	Description
CURSor:VBArs	Sets or queries the position of vertical bar cursors
CURSor:VBArs:DELTa?	Returns the difference between vbar cursors
CURSor:VBArs:POSITION<x>	Sets or queries the vbar cursor<x> horizontal position
CURSor:VBArs:POS<x>	Sets or queries the horizontal position for vertical bar cursors
CURSor:VBArs:UNIts	Sets or queries the units for vbar cursors
CURSor:WAVEform	Sets or queries the current settings for waveform cursors
CURSor:WAVEform:HDELTA?	Returns the horizontal difference between waveform cursors
CURSor:WAVEform:HPOS<x>?	Returns the position of waveform cursor <x>
CURSor:WAVEform:POSition<x>	Sets or queries the position of waveform cursor <x>
CURSor:WAVEform:STYle	Sets or queries the cursor type for waveform mode
CURSor:WAVEform:UNIts	Sets or queries the units for waveform cursors
CURSor:WAVEform:VDELTA?	Returns the vertical difference between waveform cursors
CURSor:XY?	Returns the current settings for XY cursors
CURSor:XY:PRODDELta?	Returns the product of the difference between the cursors X positions and Y positions
CURSor:XY:PRODUCT<x>?	Returns the product of the X and Y positions for the specified cursor
CURSor:XY:RADIUS<x>?	Returns the radius of the specified cursor
CURSor:XY:RATDELta?	Returns ratio of the difference between the cursors X position and Y position
CURSor:XY:RATIO<x>?	Returns ratio of the X (horizontal) and Y (vertical) position for the specified cursor
CURSor:XY:RDELta?	Returns the Dr value
CURSor:XY:READOUT	Sets or queries the XY cursor readout mode
CURSor:XY:RECTX<x>	Sets or queries the X cursor position in rectangular coordinates
CURSor:XY:RECTY<x>	Sets or queries the Y cursor position in rectangular coordinates
CURSor:XY:THDELta?	Returns the XY cursor angle delta in polar coordinates
CURSor:XY:THETA<x>?	Returns the XY cursor angle in polar coordinates
CURSor:XY:XDELta?	Returns the XY cursor ΔX value in rectangular coordinates
CURSor:XY:YDELta?	Returns the XY cursor ΔY value in rectangular coordinates

Diagnostics command group

The Diagnostic commands control the selection and execution of diagnostic tests.

Table 2-19: Diagnostics commands

Command	Description
DIAg:CONTROL:HALT	Enables or disables halting on first diagnostic failure
DIAg:CONTROL:LOOP	Enables or disables looping of diagnostics
DIAg:FAILURES:CLEAR	Sets and queries the clearing of pass/fail information from data structures, not the Event Log, at the start of diagnostic tests
DIAg:EXECUTE	Executes the currently selected set of diagnostics
DIAg:ITEM?	Returns all data associated with a selected menu item
DIAg:ITEM:FAILURES?	Returns the total number of failures that occurred
DIAg:ITEM:NAMe?	Returns the name of the selected menu item
DIAg:ITEM:RESULT?	Returns the results of the last test executed on this item
DIAg:ITEM:SUBITEMS?	Returns the number of subitems associated with this item
DIAg:LEVEL	Sets the current level of diagnostic test hierarchy
DIAg:LOOPS?	Returns the number of times the diagnostics were completed during the last execution
DIAg:NAMe?	Returns the subsystem name, area, and test name of the current diagnostic test
DIAg:NAMe:AREA?	Returns the selected area of the current diagnostic test
DIAg:NAMe:SUBSYS?	Returns the subsystem of the current diagnostic test
DIAg:NAMe:TEST?	Returns the name of the current diagnostic test
DIAg:NUMITEMS?	Returns the number of items on the currently selected level of test hierarchy
DIAg:RESults?	Returns a brief pass or fail status of the last test execution
DIAg:RESults:VERBose?	Returns a more explanatory message about the results of the last diagnostic execution
DIAg:SELect:ALL	Selects all available diagnostics
DIAg:SELect:AREA	Selects one of the available diagnostic areas
DIAg:SELect:LAST	Sets the last item of a group of items from the same level of test hierarchy
DIAg:SELect:SUBSYS	Selects one of the available diagnostic subsystems
DIAg:SELect:TEST	Selects one of the available diagnostic tests
DIAg:STATE	Sets the instrument operating state
DIAg:STOP	Terminates the execution of diagnostics
TEST	Selects and executes an item at any level of the test hierarchy
TEST:RESults?	Returns a brief pass or fail status of the last test execution
TEST:RESults:VERBose?	Returns a more explanatory message about the results of the last test execution
TEST:STOP	Terminates the execution of the test

Digital command group

NOTE. *Not available on some models.*

Use the commands in the Digital Command Group to acquire 24 digital signals and analyze them. The digital signals includes:

- Signals from each of the 16 data channels of its digital probes
- Signals derived from each of the four analog acquisition channels
- Signals derived from each of the four math expressions

Table 2-20: Digital commands

Command	Description
CQ<x>:THRESHold	Sets or queries the threshold for converting the specified clock/qualifier signal to digital form
DISplay:DIGital:HEIght	Sets or queries the height of the digital input waveform and the label associated with the channel
D<x>:LABEL	Sets or queries the label that appears for the specified digital input on the display
D<x>:POSition	Sets or queries the position for the specified digital input
D<x>:PROBE:ID:SERnumber?	Queries the serial number of the digital probe that provides the specified digital signal
D<x>:PROBE:ID:TYPe?	Queries the type of digital probe that provides the specified digital signal
D<x>:THRESHold	Sets or queries the threshold level for the digital signal specified
REF<x>:THRESHold	Sets or queries the comparable threshold for converting the reference signal to digital form
SELect:DALL	Sets the display state of all the digital inputs
SELect:DIGTraces:COMbination	Turns on the digital channels that have binary digits as 1
SELect:DIGTraces:LISt	Turns on the specified digital channels or returns the list of digital channels that are on
SELect:D<x>	Sets or queries the display state for the digital input specified

Display control command group

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

You can set the display of date and time; cursor, histogram, mask, and measurement readouts; measurement annotations; and the mode in which waveforms are displayed.

There are six color palettes from which you can select:

- **Normal.** Displays hues and lightness levels for best overall viewing.
- **Temp.** Displays areas of the waveform with the highest sample density in warmer colors (red shades) and the areas of lowest sample density in cooler colors (blue shades).
- **Spectral.** Displays areas of the waveform with the highest sample density in blue shades and the areas of lowest sample density in red shades.
- **Green.** Displays waveforms in shades of green. Areas of the waveform with the highest sample density appear in lighter green shades and the areas of lowest sample density appear in darker green shades.
- **Gray.** Displays waveforms in shades of gray. Areas of the waveform with the highest sample density appear in lighter gray shades and the areas of lowest sample density appear in darker gray shades.
- **User.** Allows you to create a customized color palette.

Use the commands to set the style that best displays your waveforms and graticule display properties.

NOTE. *The mode you choose globally affects all displayed waveforms.*

Table 2-21: Display control commands

Command	Description
DISplay?	Returns current display settings
DISplay:CLOCk	Sets or queries the display of the date/time stamp
DISplay:COLOr?	Returns color group settings
DISplay:COLOr:MATHCOLOr	Sets or queries the color to be used for math traces
DISplay:COLOr:PALEtte:IMAGEView	Sets or queries the color palette for imageview waveforms
DISplay:COLOr:PALEtte:RECORDView	Sets or queries the color palette for recordview waveforms
DISplay:COLOr:PALEtte:USEr	Returns the user palette group settings
DISplay:COLOr:PALEtte:USEr:CARet	Sets or queries the user caret color
DISplay:COLOr:PALEtte:USEr:CH<x>	Sets or queries the user palette channel colors
DISplay:COLOr:PALEtte:USEr:GRAticule	Sets or queries the user palette graticule

Table 2-21: Display control commands (cont.)

Command	Description
DISplay:COLOr:PALEtte:USEr:HIStogram	Sets or queries the user palette histogram color
DISplay:COLOr:PALEtte:USEr:MASK	Sets or queries the user palette mask color
DISplay:COLOr:PALEtte:USEr:MASKHighlight	Sets or queries the user palette mask hits color
DISplay:COLOr:PALEtte:USEr:MATH<x>	Sets or queries the user palette math colors
DISplay:COLOr:PALEtte:USEr:REF<x>	Sets or queries the user palette reference colors
DISplay:COLOr:PALEtte:USEr:WAVEform	Sets or queries the user palette waveform colors
DISplay:COLOr:REFCOLOr	Sets or queries the color to be used for reference traces
DISplay:FILTer	Sets or queries the type of interpolation to use for the display
DISplay:FORMat	Sets or queries the display format
DISplay:GRAticule	Sets or queries the type of graticule that is displayed
DISplay:INTENSITy?	Returns the waveform and graticule saturation levels
DISplay:INTENSITy:BACKLight	Sets or queries the waveform backlight intensity settings
DISplay:INTENSITy:WAVEform:IMAGEView	Sets or queries the waveform saturation level for imageview waveforms
DISplay:INTENSITy:WAVEform:RECORDView	Sets or queries the waveform saturation level for recordview waveforms
DISplay:PERSistence	Sets or queries display persistence setting
DISplay:PERSistence:RESET	Clears the persistence data
DISplay:SCREENTExt?	Returns all screen text settings
DISplay:SCREENTExt:LABel<x>?	Sets or queries the screen text setting for a given label
DISplay:SCREENTExt:LABel<x>:FONTCOlor	Sets or queries the screen text label font color
DISplay:SCREENTExt:LABel<x>:FONTNAme	Sets or queries the screen text label font name for a given label
DISplay:SCREENTExt:LABel<x>:FONTSIze	Sets or queries the screen text label font size for a given label
DISplay:SCREENTExt:LABel<x>:FONTSTyle	Sets or queries the screen text label font style for a given label
DISplay:SCREENTExt:LABel<x>:NAMe	Sets the text to be displayed for a given label
DISplay:SCREENTExt:LABel<x>:STATE	Sets or queries the screen text label state for a given label
DISplay:SCREENTExt:LABel<x>:XPOS	Sets or queries the horizontal position of a given label
DISplay:SCREENTExt:LABel<x>:YPOS	Sets or queries the vertical position of a given label
DISplay:SCREENTExt:STATE	Sets or queries the state of the display of screen text
DISplay:SHOWREmote	Sets or queries the state of the remote display feature and is equivalent to selecting Display Remote from the Display menu
DISplay:STYle	Sets or queries data display style
DISplay:TRIGBar	Sets or queries the display setting of the trigger level indicator bar(s)
DISplay:TRIGT	Sets or queries the display of the trigger point indicator
DISplay:VARpersist	Sets or queries the persistence decay time
DISplay:WAVEform	Sets or queries the display of waveform traces

E-mail command group

Commands in the E-mail group allow you to send e-mail to one or more designated recipients whenever a selected event, such as a trigger, a mask test failure (Option MTM only), or a limit test failure occurs in the instrument.

Using this feature, you do not have to continually monitor the instrument for the event.

When an event occurs, the instrument will send an SMTP mail message to one or more designated recipients through the specified mail server. The message can include any of the following:

- Screen image
- Waveform data
- Measurement data at the time of the event

NOTE. *Your instrument must be connected to the Local Area Network (LAN) using Transfer C Protocol/Internet Protocol (TCP/IP). The recipient must also have an SMTP mail server. To send e-mail for a mask test failure, you must have Mask Testing (Option MTM) installed.*

Table 2-22: E-mail commands

Command	Description
EMail	Sends a test e-mail message or sets the current e-mail sent count to zero
EMail:ATTempts	Sets or queries the number of times that an attempt will be made to send e-mail to the SMTP e-mail server
EMail:AUTHLogin	Sets or queries the login name that will be used if the SMTP e-mail server requires one for authentication
EMail:AUTHPassword	Sets the password that will be used if the SMTP e-mail server requires one for authentication
EMail:COUNt?	Returns the number of e-mails that have been sent since Email on Event was armed
EMail:FROm	Sets or queries the From line in the e-mail
EMail:HOSTwanted	Sets or queries the hostname that will be used when e-mail is sent to the SMTP e-mail server
EMail:IMAGe	Sets or queries whether image data is included in the e-mail
EMail:MASK	Sets or queries whether e-mail is sent when a mask test failure occurs
EMail:MAXSize	Sets or queries the maximum size (in megabytes) of e-mail that can be sent to the SMTP server
EMail:MEASUrement	Sets or queries whether measurement data is included in the e-mail
EMail:NUMEMails	Sets or queries the number e-mails that can be sent when Email onEvent is armed
EMail:SMTPPort	Sets or queries the SMTP port number that the e-mail server uses
EMail:SMTPServer	Sets or queries the address of the SMTP mail server

Table 2-22: E-mail commands (cont.)

Command	Description
EMail:STATUS?	Queries the status of e-mail
EMail:TIMEOut	Sets or queries the global timeout in seconds
EMail:TO	Sets or queries the address of the recipient(s) of an e-mail. Multiple addresses are separated with a semicolon (;)
EMail:TRIGger	Sets or queries whether e-mail is sent when a trigger occurs
EMail:WAVEform	Sets or queries whether waveform data is included in the e-mail

Error Detector command group

NOTE. *Not available on some models.*

The error detector provides error detection on many serial standards. This section provides error detector theory and a list of commands in the error detector command group.

Error Detector theory of operation

The Tektronix error detector option provides the following:

- Automated receiver testing using AWG7000 Series
- SATA Bit Error Detection: 1.5 Gb/s, 3.0 Gb/s, and 6.0 Gb/s Standards
- SATA Frame Error Detection: 1.5 Gb/s, 3.0 Gb/s, and 6.0 Gb/s Standards
- USB 3.0 Symbol Error Detection: 5 Gb/s Standard
- PCIe Gen1 and Gen2 Bit Error Detection: 5 Gb/s Standard
- Illegal Character and Disparity Error Detection: 1.25 Gb/s to 6.25 Gb/s
- Tracking for Spread Spectrum Clocks with up to 5000 ppm (downspread)

The following table lists the various types of error detection:

Table 2-23: Types of error detection

Error Type	(Units) Error Rate	(Units) Resolution	Effectiveness	Limitations
Character	Number of character errors per number of characters tested	Character	90% to 95%	Can miss errors if a character is changed to a legal character
Frame	Number of frame errors per number of frames tested	Frame	100% (finds all errors)	Many bit errors can equal one frame error
Symbol	Number of symbol errors per number of symbols tested	Symbol	100% (finds all errors)	Several bit errors can equal one symbol error
Bit	Number of bit errors per number of bits tested	Bit	100% (finds all errors)	Pattern length limited by memory depth

For non-USB bit error testing, you must provide the signal test pattern length and sync pattern. Align characters must be provided for all test types. All of these parameter values depend on the actual signal test pattern. The signal test pattern length is generally known from the signal generator setup.

The align character is a 10-bit character with both RD+ and RD– disparity values and must actually be present in the signal test pattern to keep the receiver aligned to the 10-bit character boundaries.

The sync pattern is a unique 10-, 20-, 30-, or 40-bit sequence (one to four 10-bit characters) that must actually be present in the signal test pattern to align the

signal so that bit-by-bit comparisons can be made. The sync pattern does not have to be “at the beginning” of the signal test pattern but it can be anywhere in the signal test pattern, as the fixed-length signal test pattern is sent repetitively by the signal generator.

To better understand the bit error detector, examine and use the setup files that contain the parameter settings in GPIB command format. Signal test patterns are supported with the following setup files:

- Any8B10BCJTPatBitErrorSetup.txt
- Any8B10BCJTPcharacterErrorSetup.txt
- PCIeCJTPatBit ErrorSetup.txt
- SataCJTPatBitErrorSetup.txt
- SataCJTPatCharactErrorSetup.txt
- SataFrameComplianceFrameErrorSetup.txt
- SataHFTPbitErrorSetjup.txt
- SataLBPbitErrorSetup.txt
- SataLFTPbitErrorSetup.txt
- SataMFTPbitErrorSetup.txt
- UsbCP0_SKPbitErrorSetup.txt
- UsbCP0_SKPcharacterErrorSetup.txt
- UsbCP0_SKPsymbolErrorSetup.txt
- PcieGen1ComplianceBitErrorSetup.txt
- PcieGen2ComplianceBitErrorSetup.txt

The oscilloscope error detector has the ability to “learn” the pattern from the signal test pattern. This capability can be used when you suspect that your DUT might have unintentional design/logic errors that modify the signal test pattern before it reaches the oscilloscope. In this case, the oscilloscope can “learn” the Modified signal test pattern, so that testing can continue even in the presence of the design/logic error.

The USB bit error and USB symbol error detection follow the same concept as SATA frame error detection. In frame error detection, the received serial data is kept aligned to 10-bit boundaries, converted from 10-bit characters to 8-bit characters, monitored for Start of Frame (SOF) and End of Frame (EOF) symbols, scrambled per the specification, to both compute the Cyclic Redundancy Check (CRC) and extract the transmitted CRC from the data. The align primitives can be detected and rejected if desired.

Examples of programmatic interface (PI) commands using SATA frame error and PCIeGen2 bit error

Example 1. A 6 Gb/s SATA Gen 3 Frame Error compliance pattern is injected into the oscilloscope Ch1. Setup Error Detector to detect Frame Errors using the following PI commands:

*RST //Establish Default Setup (wait 3 seconds for command to complete)

ERRORD:PRESET SATA3_FRAME //Select Preset Setup

ERRORD:PRESET:APPLY //Apply Preset (Applying Preset recalls the SATA3 frame error setup) (wait 1 second)

ERRORD:FRAME:TEST SYNC //Sync Recovered Clock (wait 8 seconds for command to complete)

ERRORD:FRAME:TEST:STATUS? //Status must be LOCKED to continue (If Status is not LOCK, retry SYNC command)

ERRORD:FRAME:TEST START //Start Testing for Frame Errors. (Wait 1 second)

ERRORD:FRAME:TEST:STATUS? //View Status

ERRORD:FRAME:TEST:RESULTS? //View Results (repeat the last two commands as desired)

Individual Status and Result values also may be queried. If you want to see the individual PI commands that comprise the SATA3_FRAME Preset, view the following (Win7) text file:

C:\Users\Public\Tektronix\TekScope\ErrorDetector\SataGen3FrameErrorSetup.txt

You can save and recall your own setups using the CUSTOM Preset setting.

Sample AWG setup files are also included on the oscilloscope in

C:\Users\Public\Tektronix\TekScope\ErrorDetector\AWG

To stop testing, clear test results, and quit detecting errors use these commands:

ERRORD:FRAME:TEST STOP //Stop Testing. (Automated Stop Conditions also may be defined)

ERRORD:FRAME:TEST CLEAR //Clear Test Results.

ERRORD:STATE OFF //Disable Error Detection

Example 2. A 5 Gb/s PCIE Gen2 compliance pattern is injected into the oscilloscope Ch1. Setup Error Detector to detect Bit Errors:

*RST //Establish Default Setup (wait 3 seconds for command to complete)

ERRORD:PRESET PCIE2_COMP_BIT //Select Preset Setup

ERRORD:PRESET:APPLY //Apply Preset (Applying Preset recalls the PCIE2 bit error setup) (wait 1 second)

ERRORD:BIT:TEST SYNC //Sync Recovered Clock (wait 8 seconds for command to complete)

ERRORD:BIT:TEST:STATUS? //Status must be SYNC to continue (If Status is not SYNC, retry ERRORD:BIT:TEST SYNC command)

ERRORD:BIT:TEST LEARN //Learn test pattern from signal. (wait 1 second: Once learned, the pattern remains in memory)

ERRORD:BIT:TEST START //Start Testing for Bit Errors. (wait 1 second)

ERRORD:BIT:TEST:STATUS? //Query Status

ERRORD:BIT:TEST:RESULTS? //Query Results (repeat the last two commands as desired)

The PCIE2_COMP_BIT Preset text file is located at

C:\Users\Public\Tektronix\TekScope\ErrorDetector\PcieGen2ComplianceBitErrorSetup.txt

To stop testing, clear test results, and quit detecting errors use these commands:

ERRORD:BIT:TEST:STOP //Stop Testing.

ERRORD:BIT:TEST CLEAR //Clear Test Results.

ERRORD:STATE OFF //Disable Error Detection

Table 2-24: Error Detector commands

Command	Description
General purpose group	
ERRORDetector:ALERT	Sets or queries the audible alert setting {OFF, ON} for error detection.
ERRORDetector:ALIGNPRIMitive:MINus<x>?	This command queries the align primitive minus value.
ERRORDetector:ALIGNPRIMitive:PLUS<x>?	This command queries the align primitive plus value.
ERRORDetector:BITRate	Sets or queries the bit rate enumerated value {CUSTOM, RATE312, RATE1250, RATE1500, RATE2125, RATE2500, RATE3000, RATE3125, RATE4250, RATE5000, RATE6000, RATE6250} for error detection
ERRORDetector:BITRate:VALue	Sets or queries the custom bitrate value {200000000..350000000, 1.25e9..6.25e9} for error detection. Some inetermediate values not supported. See data sheet.
ERRORDetector:CHANnel	Sets or queries the channel source {CH1..CH4} for error detection.
ERRORDetector:DURATION:COUNt	Sets or queries the test duration count as number of bits, frames, symbols, or characters to be tested for error testing
ERRORDetector:DURATION:SECOnds	Sets or queries the test duration in seconds for error testing
ERRORDetector:DURATION:TIME	Sets or queries the test duration time in days, hours, minutes, and seconds for error detector
ERRORDetector:DURATION:TIME:DAYS	Sets or queries the test duration time days component for error testing
ERRORDetector:DURATION:TIME:HOURS	Sets or queries the test duration time hours component for error testing
ERRORDetector:DURATION:TIME:MINUTES	Sets or queries the test duration time minutes component for error testing

Table 2-24: Error Detector commands (cont.)

Command	Description
ERRORDetector:DURATION:TIME:SECOnds	Sets or queries the test duration time seconds component for error testing
ERRORDetector:ERRORLIMIT	Sets or queries the error limit value to use when STOPWHEN is ERROR
ERRORDetector:FONTSIze	Sets or queries the send fontsize for the error detector UI {OFF, ON} for error detection.
ERRORDetector:PATTERNNAME	Sets or queries the pattern name stored in the setup file.
ERRORDetector:PREset	Sets or queries the preset list selection {SATA1_CJTPAT_BIT, SATA2_CJTPAT_BIT, SATA3_CJTPAT_BIT, SATA3_FRAME, SATA3_CHAR, SATA3_HFTP_BIT, SATA3_LBP_BIT, SATA3_LFTP_BIT, SATA3_MFTP_BIT, USB3_SYMBOL, USB3_CHAR, PCIE1_COMP_BIT, PCIE2_COMP_BIT, PRBS7_BIT, PRBS9_BIT, ANY_CJTPAT_BIT, ANY_CJTPAT_CHAR, CUSTOM_SETUP} for error detection.
ERRORDetector:PREset:APPLY	This command causes selected preset setup to be applied.
ERRORDetector:SAVEIMAGE	Sets or queries the save screen shot on error feature {OFF, ON} for error detection.
ERRORDetector:SAVEWFM	Sets or queries the save waveform on error feature {OFF, ON} for error detection.
ERRORDetector:SCRAMBLED	Sets or queries the status of the data scrambling option
ERRORDetector:SENDEMAIL	Sets or queries the send email on error feature {OFF, ON} for error detection.
ERRORDetector:SIGnaltype	Sets or queries the signal type {SATAGEN1, SATAGEN2, SATAGen3, USB3, PCIEGEN1, PCIEGEN2, ANY8B10B, PRBS7, PRBS9} for error detection
ERRORDetector:STANdard	Sets or queries the standard selection for error testing
ERRORDetector:STATE	Sets or queries the error detector state {OFF, ON}.
ERRORDetector:STATus?	Queries only the “most significant” or “summary” status of the error detector.
ERRORDetector:STOPWHEN	Sets or queries the stopping condition
ERRORDetector:TIMEformat	This command sets or queries error detector Elapsed Time Format.
ERRORDetector:TYPe	Sets or queries the error detector type
Bit error Detection group	
ERRORDetector:BIT:LENgth	Sets or queries the signal test pattern length in bits for error testing
ERRORDetector:BIT:SYNCPATtern?	Sets or queries all of the sync pattern items
ERRORDetector:BIT:SYNCPATtern:ADVanced	This command sets or queries the SYNCPATtern:ADVanced setting.
ERRORDetector:BIT:SYNCPATtern:BITString	Sets or queries the bit sync pattern bit string for bit error testing
ERRORDetector:BIT:SYNCPATtern:DISParity<x>	Sets or queries the four sync pattern Disparity elements.
ERRORDetector:BIT:SYNCPATtern:MINus<x>?	Sets or queries the RD- bit string sync pattern elements.
ERRORDetector:BIT:SYNCPATtern:PLUS<x>?	Sets or queries the RD+ bit string sync pattern elements.
ERRORDetector:BIT:SYNCPATtern:SYMBOL<x>	Sets or queries the symbolic sync pattern elements
ERRORDetector:BIT:TEST	Arguments START and STOP initiate and terminate bit error testing. Argument CLEAR zeroes bit error test results. Argument LEARN copies the test pattern from the signal to memory

Table 2-24: Error Detector commands (cont.)

Command	Description
ERRORDetector:BIT:TEST:COUNt?	Queries the bit error count for bit error testing
ERRORDetector:BIT:TEST:DURATION?	Queries the elapsed duration of symbols tested for bit error testing
ERRORDetector:BIT:TEST:MAXALIGNS?	Queries the maximum consecutive SATA align primitives or USB skip order sets for bit error testing
ERRORDetector:BIT:TEST:RATE?	Queries the calculated symbol error rate for bit error testing
ERRORDetector:BIT:TEST:RESults?	Queries all the results for bit error testing
ERRORDetector:BIT:TEST:SECOnds?	Queries the elapsed time in seconds for bit error testing
ERRORDetector:BIT:TEST:STATUS?	Queries all of the bit error test status bits
ERRORDetector:BIT:TEST:STATUS:LOCK?	Queries the LOCK status for bit error testing
ERRORDetector:BIT:TEST:STATUS:MAX_AP?	Queries the MAX_AP status for bit error testing
ERRORDetector:BIT:TEST:STATUS:SIGNAL?	Queries the SIGNAL status for bit error testing
ERRORDetector:BIT:TEST:STATUS:START?	Queries the START status for bit error testing
ERRORDetector:BIT:TEST:STATUS:SYNC?	Queries the SYNC status for bit error testing
ERRORDetector:BIT:TEST:TIME?	Queries the elapsed time (in days, hours, minutes, and seconds) for bit error testing
ERRORDetector:BIT:TEST:TIME:DAYS?	Queries the elapsed time days component for bit error testing
ERRORDetector:BIT:TEST:TIME:HOURS?	Queries the elapsed time hours component for bit error testing
ERRORDetector:BIT:TEST:TIME:MINUTES?	Queries the elapsed time minutes component for bit error testing
ERRORDetector:BIT:TEST:TIME:SECOnds?	Queries the elapsed time seconds component for bit error testing
ERRORDetector:SSC	Sets or queries the status of the spread spectrum clock tracking option
Frame error detection group	
ERRORDetector:FRAME?	Queries all frame error settings, status, and results
ERRORDetector:FRAme:EOF	Sets or queries the End of Frame for frame error testing
ERRORDetector:FRAme:INITIALCRCVALue	Sets or queries the initial CRC value for frame error testing
ERRORDetector:FRAme:SOF	Sets or queries the Start of Frame for frame error testing
ERRORDetector:FRAme:TEST	Arguments START and STOP initiate and terminate frame error testing. Argument CLEAR zeroes the frame error results
ERRORDetector:FRAme:TEST:BADCHARS?	Queries the illegal character count for frame error testing
ERRORDetector:FRAme:TEST:COUNt?	Queries the test error count for frame error testing
ERRORDetector:FRAme:TEST:DISParity?	Queries the disparity error count for frame error testing
ERRORDetector:FRAme:TEST:DURATION?	Queries the elapsed duration (in number of frames) tested by the frame
ERRORDetector:FRAme:TEST:MAXALIGNS?	Queries the maximum consecutive aligns encountered for frame error testing
ERRORDetector:FRAme:TEST:RATE?	Queries the calculated frame error rate for frame error testing
ERRORDetector:FRAme:TEST:RESults?	Queries all the results for frame error testing
ERRORDetector:FRAme:TEST:SECOnds?	Queries the elapsed duration in seconds for frame error testing
ERRORDetector:FRAme:TEST:STATUS?	Queries all of the frame error status at once

Table 2-24: Error Detector commands (cont.)

Command	Description
ERRORDetector:FRAme:TEST:STATUS:LOCK?	Queries the LOCK status for the frame error test
ERRORDetector:FRAme:TEST:STATUS:MAX_AP?	Queries the MAX_AP status for the frame error test
ERRORDetector:FRAme:TEST:STATUS:SIGNAL?	Queries the SIGNAL status for the frame error test
ERRORDetector:FRAme:TEST:STATUS:START?	Queries the START status for frame error tests
ERRORDetector:FRAme:TEST:TIME?	Queries the elapsed time (in days, hours, minutes, and seconds) for frame error testing
ERRORDetector:FRAme:TEST:TIME:DAYS?	Queries the elapsed time days component for frame error testing
ERRORDetector:FRAme:TEST:TIME:HOURS?	Queries the elapsed time hours component for frame error testing
ERRORDetector:FRAme:TEST:TIME:MINUTES?	Queries the elapsed time minutes component for frame error testing
ERRORDetector:FRAme:TEST:TIME:SECOnds?	Queries the elapsed time seconds component for frame error testing
Symbol error detection group	
ERRORDetector:SYMBOL?	Queries all symbol error settings, status, and results
ERRORDetector:SYMBOL:TEST	Arguments START and STOP initiate and terminate symbol error testing. Argument CLEAR zeroes the symbol error results. Argument SYNC re-syncs the recovered clock.
ERRORDetector:SYMBOL:TEST:BADCHARS?	Queries the illegal character count for symbol error testing
ERRORDetector:SYMBOL:TEST:BITCOUNT?	Queries the bit error count (number of bad bits) for symbol error testing
ERRORDetector:SYMBOL:TEST:BITDURATION?	Queries the elapsed duration (in units of bits) tested for symbol error testing
ERRORDetector:SYMBOL:TEST:BITRate?	Queries the calculated bit error rate for symbol error testing
ERRORDetector:SYMBOL:TEST:COUNt?	Queries the symbol error count for symbol error testing
ERRORDetector:SYMBOL:TEST:DISParity?	Queries the disparity error count for symbol error testing
ERRORDetector:SYMBOL:TEST:DURATION?	Queries the elapsed duration for symbol error testing
ERRORDetector:SYMBOL:TEST:MAXALIGNS?	Queries the maximum consecutive skip order sets encountered for symbol error testing
ERRORDetector:SYMBOL:TEST:RATE?	Queries the calculated symbol error rate for symbol error testing
ERRORDetector:SYMBOL:TEST:RESults?	Queries all of the results for symbol error testing
ERRORDetector:SYMBOL:TEST:SECOnds?	Queries the elapsed duration time in seconds for symbol error testing
ERRORDetector:SYMBOL:TEST:STATUS?	Queries all of the status for the symbol error tests
ERRORDetector:SYMBOL:TEST:STATUS:LOCK?	Queries the LOCK status for the symbol error test

Table 2-24: Error Detector commands (cont.)

Command	Description
ERRORDetector:SYMBOL:TEST:STATUS:MAX_AP?	Queries the MAX_AP status for the symbol error test
ERRORDetector:SYMBOL:TEST:STATUS:SIGNAL?	Queries the SIGNAL status for the symbol error test
ERRORDetector:SYMBOL:TEST:STATUS:START?	Queries the START status for symbol error testing
ERRORDetector:SYMBOL:TEST:TIME?	Queries the elapsed time (in days, hours, minutes, and seconds) for symbol error testing
ERRORDetector:SYMBOL:TEST:TIME:DAYS?	Queries the elapsed time days component for symbol error testing
ERRORDetector:SYMBOL:TEST:TIME:HOURS?	Queries the elapsed time hours component for symbol error testing
ERRORDetector:SYMBOL:TEST:TIME:MINUTES?	Queries the elapsed time minutes component for symbol error testing
ERRORDetector:SYMBOL:TEST:TIME:SECOnds?	Queries the elapsed time seconds component for symbol error testing
Character error detection group	
ERRORDetector:CHARacter?	Queries all character error settings, status, and results
ERRORDetector:CHARacter:TEST	Argument START and STOP initiate and terminate character error testing. Argument CLEAR zeroes the character error results
ERRORDetector:CHARacter:TEST:COUNt?	Queries the number of illegal characters in the signal for error testing
ERRORDetector:CHARacter:TEST:DISPCOUNT?	Queries the number of disparity errors in the signal for character error testing
ERRORDetector:CHARacter:TEST:DISPRATE?	Queries the calculated disparity error rate for character error testing
ERRORDetector:CHARacter:TEST:DURATION?	Queries the elapsed duration in units of characters tested
ERRORDetector:CHARacter:TEST:MAXALIGNS?	Queries the maximum consecutive aligns encountered for character error testing
ERRORDetector:CHARacter:TEST:RATE?	Queries the calculated illegal character error rate for error testing
ERRORDetector:CHARacter:TEST:RESults?	Queries all of the character error test results
ERRORDetector:CHARacter:TEST:SECOnds?	Queries the elapsed duration time seconds component for character error testing
ERRORDetector:CHARacter:TEST:STATUS?	Queries all of the status for character error tests
ERRORDetector:CHARacter:TEST:STATUS:LOCK?	Queries the LOCK status for the character error test
ERRORDetector:CHARacter:TEST:STATUS:MAX_AP?	Queries the MAX_AP status for the character error test
ERRORDetector:CHARacter:TEST:STATUS:SIGNAL?	Queries the SIGNAL status for the character error test

Table 2-24: Error Detector commands (cont.)

Command	Description
ERRORDetector:CHARacter:TEST:STATUS:START?	Queries the START status for the character error testing
ERRORDetector:CHARacter:TEST:TIME?	Queries the elapsed time (in days, hours, minutes, and seconds) for character error testing
ERRORDetector:CHARacter:TEST:TIME:DAYS?	Queries the elapsed time days component for character error testing
ERRORDetector:CHARacter:TEST:TIME:HOURS?	Queries the elapsed time hours component for character error testing
ERRORDetector:CHARacter:TEST:TIME:MINUTES?	Queries the elapsed time minutes component for character error testing
ERRORDetector:CHARacter:TEST:TIME:SECOnds?	Queries the elapsed time seconds component for character error testing
File group	
ERRORDetector:FILE:RECAll	Initiates a file recall of error detector parameters from a text file
ERRORDetector:FILE:SAVe	Initiates a file save of error detector parameters to a text file
Align pattern group	
ERRORDetector:ALIGNCHARacter?	Queries both of the align character values
ERRORDetector:ALIGNCHARacter:MINus?	Sets or queries the RD- align character value
ERRORDetector:ALIGNCHARacter:PLUS?	Sets or queries the RD+ align character value
ERRORDetector:ALIGNCHARacter:SYMBOL	Sets or queries the Symbolic align character
ERRORDetector:ALIGNPRIMitive?	Queries all of the align primitive values
ERRORDetector:ALIGNPRIMitive:MINUS?	Queries the RD- align primitive enable and string values
ERRORDetector:ALIGNPRIMitive:PLUS?	Queries the RD+ align primitive enable and string values
ERRORDetector:ALIGNPRIMitive:STATE	Sets or queries the align primitive enable
ERRORDetector:ALIGNPRIMitive:SYMBOL<x>	Sets or queries the align primitive symbolic values
ERRORDetector:MAXALIGNS	Sets or queries the maximum consecutive align primitives before an error
Skip order set group	
ERRORDetector:SKIPSETPRIMitive:MINus<x>?	Queries the RD- skip order set array of bit string values.
ERRORDetector:SKIPSETPRIMitive:PLUS<x>?	This command queries the skip order RD+ array of bit string values.
ERRORDetector:SKIPSETPRIMitive:STATE	Sets or queries the status of the RD+ skip order set option
ERRORDetector:SKIPSETPRIMitive:SYMBOL<x>	Sets or queries the symbolicc skip order set values

File system command group

Use the commands in the File System Command Group to help you use the built-in hard disk drive. You can use the commands to do the following:

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

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

- File arguments are always enclosed within double quotes: “C:\MYDIR\TEK00001.SET”
- File names follow the MS-DOS format: [DRIVE:][\PATH\]filename
- Path separators can be either forward slashes (/) or back slashes (\)

NOTE. *Using a back slash as a path separator can produce some unexpected results, depending on how your controller application treats escaped characters. Many applications recognize the sequence of a 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” might be interpreted as a newline character; “\t” might be interpreted as a tab character. To ensure that this interpretation does not occur, you can use double back slashes. For example, “C:\\testfile.txt”.*

- Some FILESystem commands can fail because a file has read-only attributes. You will not be able to delete or replace such files until this attribute is removed. Refer to the operating system help on file properties for further information.

Table 2-25: File system commands

Command	Description
FILESystem?	Returns the file system state
FILESystem:COPy	Copies one or more files to a new file
FILESystem:CWD	Sets or queries the current working directory for FILESystem GPIB commands
FILESystem:DELEte	Deletes a named file or directory
FILESystem:DIR?	Returns a list of directory contents
FILESystem:MKDir	Makes a new directory
FILESystem:PRInt	Prints a named file to the named port
FILESystem:READFile	Copies the named file to the GPIB port
FILESystem:REName	Assigns a new name to an existing file
FILESystem:RMDir	Deletes the named directory
FILESystem:WRITEFile	Copies the GPIB port block data to a named file

Hard copy command group

Hard Copy commands enable you to make hard copies of data files or send the data to a specified file. In addition, these commands provide you with information about (and the ability to set) file path settings.

Table 2-26: Hard copy commands

Command	Description
EXPort	Copies a waveform to a specified file or returns formatting and file information
EXPort:FILEName	Sets or queries the export file path
EXPort:FORMat	Sets or queries the export image format
EXPort:PALEtte	Sets or queries the export color palette
EXPort:READOuts	Sets or queries the position of the readouts for export
EXPort:VIEW	Sets or queries the export screen capture area
HARDCopy	Sends a screen copy to the selected port or returns the selected port and file path
HARDCopy:FILEName	Sets or queries the hard copy file path
HARDCopy:LAYout	Sets or queries the page orientation for hard copy
HARDCopy:PALEtte	Sets or queries the hard copy color palette
HARDCopy:PORT	Sets or queries whether the hard copy data will be sent to a file or printed on the next hard copy command
HARDCopy:READOuts	Sets or queries the position of the readouts for hardcopy
HARDCopy:VIEW	Sets or queries the area of the screen to be hardcopied

Histogram command group

Histogram commands let you select the type of histogram, specify what part of the waveform should go into the histogram, and retrieve histogram statistics. Use commands from this group to do the following:

- Create a histogram of vertical or horizontal values for a channel, math, or reference waveform
- Adjust the limits of the waveform area from which the histogram data is obtained
- Clear histogram count and restart
- Retrieve the histogram data

Table 2-27: Histogram commands

Command	Description
HIStogram?	Returns all histogram parameters
HIStogram:BOX	Sets or queries the left, top, right, and bottom positions of the histogram box, in source waveform coordinates
HIStogram:BOXPcnt	Sets or queries same as `HIStogram:BOX`, but in percentage coordinates, with 0,0 upper left and 100,100 lower right
HIStogram:COUNt	Clears histogram count source data and restarts counting
HIStogram:DATa?	Returns a comma-separated list of histogram data numbers
HIStogram:DISplay	Sets or queries whether histogram data is displayed and the current display type setting
HIStogram:FUNCtion	Sets the type of histogram to create Or queries the current histogram type
HIStogram:MODe	Sets type of histogram to create or turns histograms off Or queries the current histogram type or that histogram display is disabled
HIStogram:SIZe	Sets or queries the width (or height) of the histogram on the screen in divisions
HIStogram:SOUrce	Sets or queries which source waveform will be compared against the histogram box when the histogram testing is enabled
HIStogram:STATE	Sets or queries whether histogram calculations are enabled

Horizontal command group

Horizontal commands control the time bases of the instrument. You can set the time per division (or time per point) of the main time base. You can use the Horizontal commands to do the following:

- Set the scale, horizontal position and reference, and units of the time base
- Get the screen resolution, time of first point and time of last point, or get all the horizontal settings
- Enable or disable the display of the time base

You can substitute SECdiv for SCAle in the horizontal commands. This provides program compatibility with earlier models of Tektronix instruments.

Table 2-28: Horizontal commands

Command	Description
HORizontal?	Returns all learnable settings for the horizontal commands
HORizontal:ACQDURATION?	Returns the time base duration
HORizontal:ACQLENGTH?	Returns the record length
HORizontal:DIGital:MAGnivu:POSition?	Queries the horizontal position of the MagniVu digital acquisition ***NOTE.*** *Not available on some models*
HORizontal:DIGital:RECOrdlength?	Queries the record length of the digital acquisition
HORizontal:DIGital:RECOrdlength:MAGnivu?	Queries the record length of the MagniVu digital acquisition ***NOTE.*** *Not available on some models*
HORizontal:DIGital:RECOrdlength:MAIn?	Queries the record length of the main digital acquisition
HORizontal:DIGital:SAMPLERate?	Queries the sample rate of the digital acquisition
HORizontal:DIGital:SAMPLERate:MAGnivu?	Queries the sample rate of the MagniVu digital acquisition ***NOTE.*** *Not available on some models*
HORizontal:DIGital:SAMPLERate:MAIn?	Queries the sample rate of the main digital acquisition
HORizontal:DIVisions?	Returns the number of graticule divisions over which the waveform is displayed
HORizontal:FASTframe?	Returns all settings for the horizontal FastFrame commands
HORizontal:FASTframe:COUNt	Sets or queries the FastFrame frame count
HORizontal:FASTframe:MAXFRames?	Returns the maximum number of FastFrame frames which can be acquired at the current frame length
HORizontal:FASTframe:MULtipleframes:FRAMESTart:<wfm>	Sets or queries the start frame number on the specified waveform for FastFrame multiple frames
HORizontal:FASTframe:MULtipleframes:MODe	Sets or queries the mode for FastFrame multiple frames
HORizontal:FASTframe:MULtipleframes:NUMFRames:<wfm>	Sets or queries the number of frames on the specified waveform for FastFrame multiple frames
HORizontal:FASTframe:REF:FRAme	Sets or queries the FastFrame reference frame number
HORizontal:FASTframe:REF:SOUrce	Sets or queries the FastFrame Reference waveform source

Table 2-28: Horizontal commands (cont.)

Command	Description
HORizontal:FASTframe:SELECTED:CH<x>	Sets or queries the FastFrame selected frame number on the specified waveform
HORizontal:FASTframe:SELECTED:MATH<x>	Sets or queries the FastFrame selected frame number on the specified waveform
HORizontal:FASTframe:SELECTED:REF<x>	Sets or queries the FastFrame selected frame number on the specified waveform
HORizontal:FASTframe:SELECTED:SOUrce	Sets or queries the FastFrame source waveform
HORizontal:FASTframe:SEQuence	Sets or queries the FastFrame source waveform
HORizontal:FASTframe:SINGLEFramemath	Sets or queries FastFrame single frame math
HORizontal:FASTframe:SIXteenbit	Sets or queries FastFrame sixteen bit
HORizontal:FASTframe:STATE	Sets or queries the state of FastFrame acquisition
HORizontal:FASTframe:SUMFrame	Sets or queries the summary frame mode
HORizontal:FASTframe:TIMEStamp:ALL:<wfm>?	Returns the frame number and time stamp for each frame between requested frames, inclusive, within the specified waveform
HORizontal:FASTframe:TIMEStamp:BETWeen:<wfm>?	Returns the relative trigger for the delta time between the specified frames, within the specified waveform
HORizontal:FASTframe:TIMEStamp:DELTa:<wfm>?	Returns the relative time between the triggers of the FastFrame Selected and the FastFrame Reference, within the specified waveform
HORizontal:FASTframe:TIMEStamp:FRAMe:<wfm>?	Returns the absolute trigger date and time for the specified frame and waveform
HORizontal:FASTframe:TIMEStamp:REF?	Returns the absolute trigger date and time for the FastFrame specified reference waveform
HORizontal:FASTframe:TIMEStamp:SELECTED:<wfm>?	Returns the absolute trigger date and time for the FastFrame Selected, within the specified waveform
HORizontal:FASTframe:TRACk	Sets up or returns the state of FastFrame tracking
HORizontal:FASTframe:XZEro:ALL:CH<x>?	Returns the time from the trigger to the start of the specified frames on the specified channel
HORizontal:FASTframe:XZEro:ALL:REF<x>?	Returns the time from the trigger to the start of the specified frames on the specified reference
HORizontal:FASTframe:XZEro:FRAme:CH<x>?	Returns the time from the trigger to the start of the specified frame on the specified channel
HORizontal:FASTframe:XZEro:FRAme:REF<x>?	Returns the time from the trigger to the start of the specified frame on the specified reference
HORizontal:FASTframe:XZEro:REF?	Sets or queries the time from the trigger to the trigger sample on the reference waveform
HORizontal:FASTframe:XZEro:SELECTED:CH<x>?	Sets or queries the time from the trigger to the trigger sample on the selected channel
HORizontal:FASTframe:XZEro:SELECTED:REF<x>?	Sets or queries the time from the trigger to the trigger sample on the selected reference waveform
HORizontal:MAIn?	Returns the time per division of the main time base
HORizontal[:MAIn]:DELay:MODe	Sets or queries the main time base trigger delay mode

Table 2-28: Horizontal commands (cont.)

Command	Description
HORizontal[:MAIn]:DELay:POSition	Sets or queries the main time base position when Horizontal Delay Mode is turned off
HORizontal[:MAIn]:DELay:TIMe	Sets or queries the main time base trigger delay time
HORizontal:MAIn:INTERPRatio?	Returns the main horizontal time base interpolation ratio
HORizontal[:MAIn]:POSition	Sets or queries the waveform horizontal position on the display
HORizontal:MAIn:UNIts	Sets or queries the units for the horizontal time base
HORizontal:MAIn:UNIts:STRing	Sets or queries the units string for the horizontal main time base trigger delay
HORizontal:MODE	Sets or queries the horizontal mode. Auto mode is the factory default
HORizontal:MODE:AUTO:LIMITrecordlen	Sets or queries the record length limit used by the auto horizontal mode
HORizontal:MODE:RECOrdlength	Sets or queries the record length
HORizontal:MODE:SAMPLERate	Sets or queries the sample rate
HORizontal:MODE:SCAle	Sets or queries the horizontal scale
HORizontal:POSition	Either positions the waveform horizontally on the display or returns its horizontal position on the display
HORizontal:ROLL	Sets or queries the horizontal roll mode status
HORizontal:TIMEStamp:CH<x>?	Returns the absolute trigger date and time for the specified frame and waveform
HORizontal:TIMEStamp:REF<x>?	Returns the absolute trigger date and time for FastFrame reference

Limit Test command group

Use the commands in the Limit Test Command Group to perform limit testing by selecting channel, math, or reference waveforms and comparing them against a template waveform.

Table 2-29: Limit Test commands

Command	Description
LIMit?	Returns all settings for the Limit commands
LIMit:BEEP	Causes the instrument to beep when the waveform data exceeds the limits set in the limit test
LIMit:COMpare	Resets the limit test comparison template
LIMit:COMpare:CH<x>	Sets or queries the template against which to compare the waveform acquired from the channel specified by x
LIMit:COMpare:MATH<x>	Sets or queries the template against which to compare the math waveform specified by x
LIMit:COMpare:REF<x>	Sets or queries the template against which to compare the reference waveform specified by x
LIMit:EMail	Sets or queries whether an E-mail is generated when the source waveform data exceeds the limits specified for the limit test
LIMit:HARDCopy	Sets or queries whether a hard copy operation is executed on the waveform when any waveform data exceeds the limit set in the limit test. LIMit:STATE must be set to ON for the hard copy operation to execute
LIMit:HIGHLIGHTHits	Sets or queries whether violation highlighting occurs when limit testing is active, and, if the RESET argument is set, clears the highlighting
LIMit:HIGHLIGHTHits:RESet	Resets the hits highlighting for limit testing
LIMit:LOCk	Sets or queries whether vertical scaling and positioning affect both source and template for template comparison pairs
LIMit:LOG	Sets or queries whether a log file is saved when the source waveform data exceeds the test limits
LIMit:SAVEWFM	Sets or queries whether the source waveform is saved when the source waveform data exceeds the test limits
LIMit:SAVEWFM:FILEName	Sets or queries the path where waveforms or log files will be saved when the waveform data exceeds the limits set by the limit test
LIMit:SRQ	Sets or queries whether a Service Request Interrupt (SRQ) is generated when the waveform data falls outside of the test limits
LIMit:STATE	Sets limit testing on or off or queries whether limit testing is in effect
LIMit:STATus?	Queries the state of limit testing
LIMit:STOPOnviolation	Sets or queries whether acquisitions are stopped when the waveform data exceeds the test limits
LIMit:TEMPlate:STORe	Saves the specified source waveform to the specified reference or file name
LIMit:TEMPlate:TOLerance:HORizontal	Sets or queries the amount in units of horizontal divisions, by which the source waveform is varied horizontally when creating the destination waveform

Table 2-29: Limit Test commands (cont.)

Command	Description
LIMit:TEMPlate:TOLerance:VERTical	Sets or queries the amount in units of vertical divisions, by which the source waveform is varied vertically when creating the destination waveform
LIMit:TEMPlate:DESTination	Sets or queries destination reference waveform that the limit template destination command will use
LIMit:TEMPlate:SOUrce	Sets or queries the channel, math waveform, or reference waveform that the LIMit:TEMPlate:STORe command will use

Link Training command group

Use these commands to control link training on the instrument.

Table 2-30:

Command	Description
LINKTRaining:ACQTime	sets or queries the maximum Acquire Time for Link Training.
LINKTRaining:ARMscope	Sets or queries the ARMscope setting of Link Training.
LINKTRaining:DECOde	Sets or queries the Decode setting of the Link Training.
LINKTRaining:EQUalization	Sets or queries the equalization value for Link Training.
LINKTRaining:LANE	Sets or queries the activation of the individual communications lanes under test.
LINKTRaining:MARK	Sets or queries the Mark setting of the Link Training application.
LINKTRaining:SETUP	Saves or restores custom Link Training setup files.
LINKTRaining:STANdard	Sets or queries the communication standard for Link Training.
LINKTRaining:TRIGgeron	Sets or queries the TriggerOn setting of Link Training.
LINKTRaining:STAte	Sets or queries the state of the Link Training application.

Low Speed Serial Trigger command group

Use the commands in the Low Speed Serial Trigger Command Group to control serial triggering for the instrument.

NOTE. *CAN, FlexRay, I2C, LIN, MIPI, PCIE, RS-232, SPI, USB, and 8B10B trigger commands are optional.*

Table 2-31: Low Speed Serial Trigger commands

Command	Description
TRIGger:A:BUS:CAN:CONDition	Sets or queries the CAN bus trigger condition
TRIGger:A:BUS:CAN:DATa:DIRection	Sets or queries the CAN bus trigger data direction
TRIGger:A:BUS:CAN:DATa:FORMat	Sets or queries the CAN bus trigger data format
TRIGger:A:BUS:CAN:DATa:QUALifier	Sets or queries CAN bus trigger data qualifier
TRIGger:A:BUS:CAN:DATa:SIZe	Sets or queries the CAN bus trigger data size
TRIGger:A:BUS:CAN:DATa:VALue	Sets or queries CAN bus trigger data value
TRIGger:A:BUS:CAN:FRAMEtype	Sets or queries CAN bus trigger frame type
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:DIRection	Sets or queries the CAN bus trigger identifier direction
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:FORMat	Sets or queries the CAN bus identifier format
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:MODe	Sets or queries CAN bus trigger identifier mode
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:VALue	Sets or queries the CAN bus trigger identifier value
TRIGger:A:BUS:DATa:FORMat	Sets or queries the trigger data format for the bus
TRIGger:A:BUS:DATa:VALue	Sets or queries the trigger data value depending on the format selected for the bus
TRIGger:A:BUS:FLEXRAY:CONDition	Sets or queries the FLEXRAY bus trigger condition
TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat	Sets or queries FLEXRAY bus trigger cycle count format
TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier	Sets or queries FLEXRAY bus trigger cycle count qualifier
TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue	Sets or queries the FLEXRAY bus trigger cycle count value
TRIGger:A:BUS:FLEXRAY:DATa:FORMat	Sets or queries the FLEXRAY bus trigger data format
TRIGger:A:BUS:FLEXRAY:DATa:OFFSet	Sets or queries the FLEXRAY bus trigger data offset
TRIGger:A:BUS:FLEXRAY:DATa:QUALifier	Sets or queries the FLEXRAY bus trigger data qualifier
TRIGger:A:BUS:FLEXRAY:DATa:SIZe	Sets or queries the FLEXRAY bus trigger data size
TRIGger:A:BUS:FLEXRAY:DATa:VALue	Sets or queries the FLEXRAY bus trigger data value
TRIGger:A:BUS:FLEXRAY:EOFTYPE	Sets or queries the FLEXRAY bus end of file type
TRIGger:A:BUS:FLEXRAY:ERRTYPE	Sets or queries the FLEXRAY bus trigger error type

Table 2-31: Low Speed Serial Trigger commands (cont.)

Command	Description
TRIGger:A:BUS:FLEXRAY:FRAMEType	Sets or queries the FLEXRAY bus trigger frame type
TRIGger:A:BUS:FLEXRAY:HEADER:CRC	Sets or queries the FLEXRAY bus trigger header CRC
TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount	Sets or queries the FLEXRAY bus trigger header cycle count
TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID	Sets or queries the FLEXRAY bus trigger header frame ID
TRIGger:A:BUS:FLEXRAY:HEADER:INDBits	Sets or queries the FLEXRAY bus trigger header Indicator Bits
TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth	Sets or queries the FLEXRAY bus trigger header Payload Length
TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat	Sets or queries the FLEXRAY bus trigger identifier format
TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier	Sets or queries the FLEXRAY bus trigger identifier qualifier
TRIGger:A:BUS:FLEXRAY:IDentifier:VALue	Sets or queries the FLEXRAY bus trigger identifier value
TRIGger:A:BUS:I2C:ADDress:FORMat	Sets or queries the address format for the I2C bus
TRIGger:A:BUS:I2C:ADDress:MODe	Sets or queries the address mode for the I2C bus
TRIGger:A:BUS:I2C:ADDress:TYPe	Sets or queries the address type for the I2C bus trigger
TRIGger:A:BUS:I2C:ADDress:VALue	Sets or queries the binary address string used for the I2C trigger of the bus
TRIGger:A:BUS:I2C:CONDition	Sets or queries the trigger condition for the I2C bus
TRIGger:A:BUS:I2C:DATa:DIRection	Sets or queries the data direction for the I2C bus
TRIGger:A:BUS:I2C:DATa:FORMat	Sets or queries the trigger data format for the I2C bus
TRIGger:A:BUS:I2C:DATa:SIZe	Sets or queries the length of the data string in bytes to be used for an I2C trigger
TRIGger:A:BUS:I2C:DATa:VALue	Sets or queries the data value of the data token for an I2C trigger
TRIGger:A:BUS:LIN:CONDition	Sets or queries the LIN bus trigger condition
TRIGger:A:BUS:LIN:DATa:FORMat	Sets or queries the LIN bus trigger data format
TRIGger:A:BUS:LIN:DATa:QUALifier	Sets or queries the LIN bus trigger data qualifier
TRIGger:A:BUS:LIN:DATa:SIZe	Sets or queries the LIN bus trigger data size
TRIGger:A:BUS:LIN:DATa:VALue	Sets or queries the LIN bus trigger data value
TRIGger:A:BUS:LIN:ERRTYPE	Sets or queries the LIN bus trigger error type
TRIGger:A:BUS:LIN:IDentifier:FORMat	Sets or queries the LIN bus trigger identifier format
TRIGger:A:BUS:LIN:IDentifier:VALue	Sets or queries the LIN trigger identifier value
TRIGger:A:BUS:PCIE:CHARacter:CHAR	Sets or queries the PCIE bus trigger character
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus	Sets or queries the PCIE bus trigger character symbol minus
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS	sets or queries PCIE bus trigger character symbol plus
TRIGger:A:BUS:PCIE:CONDition	Sets or queries the PCIE bus trigger condition
TRIGger:A:BUS:PCIE:DISParity	Sets or queries the PCIE bus trigger disparity
TRIGger:A:BUS:PCIE:ERROR	Sets or queries the PCIE bus trigger error

Table 2-31: Low Speed Serial Trigger commands (cont.)

Command	Description
TRIGger:A:BUS:PCIE:FORMat	Sets or queries the PCIe bus trigger format
TRIGger:A:BUS:PCIE:PATtern:CHAR<x>	Sets or queries the PCIE bus trigger pattern character
TRIGger:A:BUS:PCIE:PATtern:ORDERedset	Sets or queries the PCIE bus trigger pattern ordered set
TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>	Sets or queries the PCIE bus trigger pattern symbol minus
TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>	Sets or queries the PCIE bus trigger pattern plus symbol
TRIGger:A:BUS:RS232C:CONDition	Sets or queries the trigger condition for the specified RS232 bus
TRIGger:A:BUS:RS232C:DATa:FORMat	Sets or queries the data format for the RS232c bus trigger
TRIGger:A:BUS:RS232C:DATa:SIZe	Sets or queries the length of the data string in bytes to be used for an RS232 trigger
TRIGger:A:BUS:RS232C:DATa:VALue	Sets or queries the data value of the data token for an RS232 trigger
TRIGger:A:BUS:SOUrce	Sets or queries the source for a bus trigger
TRIGger:A:BUS:SPI:CONDition	Sets or queries the trigger condition for the SPI bus
TRIGger:A:BUS:SPI:DATa:FORMat	Sets or queries the format of the data to be used for an SPI trigger when the trigger condition is Data
TRIGger:A:BUS:SPI:DATa:SIZe	Sets or queries the length of the data string in bytes to be used for an SPI trigger
TRIGger:A:BUS:SPI:DATa:VALue	Sets or queries the data value of the data token for an SPI trigger
TRIGger:A:BUS:USB:ADDress:FORMat	Sets or queries the address format for the USB bus trigger normal token
TRIGger:A:BUS:USB:ADDress:HIVALue	Sets address value for normal token to be used with In Range and Out of Range qualifiers
TRIGger:A:BUS:USB:ADDress:VALue	Sets address value for normal token
TRIGger:A:BUS:USB:CHARacter:CHAR	Sets or queries the USB character when the Trigger Condition is set to Character.
TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus	Sets or queries the character bit pattern (negative disparity) when the Trigger Condition is set to Character.
TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS	Sets or queries the Character bit pattern (positive disparity view) when the Trigger Condition is set to Character.
TRIGger:A:BUS:USB:CONDition	Sets or queries the trigger condition for the specified USB bus
TRIGger:A:BUS:USB:DATa:FORMat	Sets or queries the data format for the USB bus trigger
TRIGger:A:BUS:USB:DATa:HIVALue	Sets or queries the trigger data value for data token of the specified USB bus
TRIGger:A:BUS:USB:DATa:OFFSet	Sets or queries the data offset for the USB bus trigger
TRIGger:A:BUS:USB:DATa:SIZe	Sets or queries the number of contiguous data bytes for the USB bus trigger
TRIGger:A:BUS:USB:DATa:TYPe	Sets or queries the data packet type when USB bus trigger condition is Data Packet
TRIGger:A:BUS:USB:DATa:VALue	Sets or queries the USB bus trigger data value for data token
TRIGger:A:BUS:USB:DISParity	Sets or queries the type of disparity that the bus will trigger on.
TRIGger:A:BUS:USB:ENDPoint:FORMat	Sets or queries the endpoint format for data token to be used with in range and out of range qualifiers
TRIGger:A:BUS:USB:ENDPoint:HIVALue	Sets or queries the endpoint value for data token to be used with in range and out of range qualifiers
TRIGger:A:BUS:USB:ENDPoint:VALue	Sets or queries the endpoint value for normal token of the specified USB bus

Table 2-31: Low Speed Serial Trigger commands (cont.)

Command	Description
TRIGger:A:BUS:USB:ERROR	Sets or queries whether the oscilloscope will trigger on a character error or a disparity error.
TRIGger:A:BUS:USB:ERRTYPE	Sets or queries the error type when USB bus trigger on condition is set to Error
TRIGger:A:BUS:USB:FORMat	Sets or queries the pattern editing format.
TRIGger:A:BUS:USB:HANDShaketype	Sets or queries the handshake type when USB bus trigger on condition is set to Handshake Packet
TRIGger:A:BUS:USB:PATtern:CHAR<x>	Sets or queries the specified character when the Trigger Condition is set to Pattern.
TRIGger:A:BUS:USB:PATtern:ORDERedset	Sets or queries the state of triggering on an ordered set.
TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>	Sets or queries the specified character bit pattern (negative disparity) when the Trigger Condition is set to Pattern.
TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>	Sets or queries the specified character bit pattern (positive disparity) when the Trigger Condition is set to Pattern.
TRIGger:A:BUS:USB:QUAlifier	Sets or queries the USB bus trigger qualifier for address, endpoint, and data
TRIGger:A:BUS:USB:SOF:FORMat	Sets or queries the format for the SOF frame number
TRIGger:A:BUS:USB:SOF:FRAMENUMber	Sets frame number for the USB bus trigger when the condition is Start Of Frame (SOF)
TRIGger:A:BUS:USB:SPECIALType	Sets or queries the PID value for Special Packet USB bus trigger on condition
TRIGger:A:BUS:USB:SPLIT:ET:VALue	Sets or queries the Endpoint Type value of the specified USB bus trigger
TRIGger:A:BUS:USB:SPLIT:HUB:FORMat	Sets or queries the format for the hub address of the USB bus trigger on split token field
TRIGger:A:BUS:USB:SPLIT:HUB:VALue	Sets or queries the hub address for SPLIT token of the USB bus trigger
TRIGger:A:BUS:USB:SPLIT:PORT:FORMat	Sets the port address format for the USB bus trigger on split token field
TRIGger:A:BUS:USB:SPLIT:PORT:VALue	Sets or queries the port address for SPLIT token of the USB bus trigger
TRIGger:A:BUS:USB:SPLIT:SC:VALue	Sets or queries the Start/Complete value for the USB bus trigger on split token field
TRIGger:A:BUS:USB:SPLIT:SE:VALue	Sets or queries the Start/End value for the USB bus trigger
TRIGger:A:BUS:USB:TOKENType	Sets or queries the Token Type when USB Trigger condition is set to Token (Address) Packet
DPO7000 Series only	
TRIGger:A:CAN:CONDition	Sets or queries the CAN condition ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:DATa:DIRection	Sets or queries the CAN trigger condition to be valid on a READ, WRITE, or either ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:FORMat	Sets or queries the CAN data format ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:DATa:LEVel	Sets or queries the CAN Trigger threshold for the CAN data source ***NOTE.*** *This trigger is available only with Option LSA installed*

Table 2-31: Low Speed Serial Trigger commands (cont.)

Command	Description
TRIGger:A:CAN:DATa:SOUrce	Sets or queries the CAN data source ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:DATa:VALue	Sets or queries the binary data string used for CAN Trigger if the trigger condition is ID or IDANDDATA ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:FRAMEtype	Sets or queries the CAN trigger frame type ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:IDENTifier:MODe	Sets or queries the CAN trigger identifier mode ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:IDENTifier:VALue	Sets or queries the binary address string used for the CAN trigger if the trigger condition is ID or IDANDDATA ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:PROBE	Sets or queries the probing method used to probe the CAN signal ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:CAN:SPEed	Sets or queries the bit rate of the CAN system ***NOTE.*** *This trigger is available only with Option LSA installed*
TRIGger:A:I2C:ADDRess:MODe	Sets or queries the I2C address mode to 7 or 10 bit
TRIGger:A:I2C:ADDRess:RWINClude	Sets or queries the I2C address mode to include the read/write bit as an additional 8th bit of the 7-bit address
TRIGger:A:I2C:ADDRess:TYPe	Sets or queries the I2C address type
TRIGger:A:I2C:ADDRess:VALue	Sets or queries the binary address string used for the I2C trigger if the trigger condition is ADDR or ADDRANDDATA
TRIGger:A:I2C:CLOCk:LEVel	Sets or queries the clock source for the I2C Serial Trigger
TRIGger:A:I2C:CLOCk:SOUrce	Sets or queries the clock source for the I2C Serial Trigger
TRIGger:A:I2C:CONDition	Sets or queries the trigger condition for the I2C trigger
TRIGger:A:I2C:DATa:DIRection	Sets or queries the I2C trigger condition valid on a READ, WRITE, or either
TRIGger:A:I2C:DATa:LEVel	Sets or queries the threshold level for the I2C data source
TRIGger:A:I2C:DATa:SOUrce	Sets or queries the data source for the I2C serial trigger
TRIGger:A:I2C:DATa:STARt	Sets or queries the data start byte
TRIGger:A:I2C:DATa:VALue	Sets or queries the binary data string used for I2C triggering if the trigger condition is DATA or ADDRANDDATA
TRIGger:A:I2C:FORMat	Sets or queries the display format for the I2C data value
TRIGger:A:RS232:BAUd	Sets or queries the baud rate for RS232 triggering
TRIGger:A:RS232:DATa:LEVel	Sets or queries the threshold for the RS232 data source
TRIGger:A:RS232:DATa:SOUrce	Sets or queries the RS232 data source
TRIGger:A:RS232:DATa:VALue	Sets or queries the binary data string used for RS232 triggering

Table 2-31: Low Speed Serial Trigger commands (cont.)

Command	Description
TRIGger:A:RS232:FORMat	Sets or queries the display format for the RS232 data value
TRIGger:A:RS232:PARity	Sets or queries the state of the RS232 parity bit
TRIGger:A:SPI:CONDition	Sets or queries the trigger condition for SPI triggering
TRIGger:A:SPI:DATa:MISO:ACTIVE	Sets or queries the SPI MISO polarity
TRIGger:A:SPI:DATa:MISO:LEVel	Sets or queries the threshold for the SPI MISO data source
TRIGger:A:SPI:DATa:MISO:SOUrce	Sets or queries the MISO data source for the SPI trigger
TRIGger:A:SPI:DATa:MISO:VALue	Sets or queries the binary data string used for the SPI trigger if the trigger condition is set to MISO or MISOMOSI
TRIGger:A:SPI:DATa:MOSI:ACTIVE	Sets or queries the SPI MOSI polarity
TRIGger:A:SPI:DATa:MOSI:LEVel	Sets or queries the threshold for the SPI MOSI data source
TRIGger:A:SPI:DATa:MOSI:SOUrce	Sets or queries the MOSI data source for the SPI trigger
TRIGger:A:SPI:DATa:MOSI:VALue	Sets or queries the binary data string used for the SPI trigger if the trigger condition is set to MOSI or MISOMOSI
TRIGger:A:SPI:DATa:STARt	Sets or queries the nth data byte on the data source after the signal on the enable slave source switches to the polarity specified by the Slave Select Polarity
TRIGger:A:SPI:FORMat	Sets or queries the SPI trigger data format
TRIGger:A:SPI:SCLK:ACTIVE	Sets or queries the SPI SCLK polarity
TRIGger:A:SPI:SCLK:LEVel	Sets or queries the threshold for the SPI trigger SCLK
TRIGger:A:SPI:SCLK:SOUrce	Sets or queries the SPI SCLK source
TRIGger:A:SPI:SS:ACTIVE	Sets or queries the SPI trigger Slave Select (SS) polarity
TRIGger:A:SPI:SS:LEVel	Sets or queries the threshold for the SPI trigger Slave Select (SS) signal
TRIGger:A:SPI:SS:SOUrce	Sets or queries the SPI trigger Slave Select (SS) source

Mask command group

Mask commands compare incoming waveforms to standard or user-defined telecommunications masks. Standard masks are available only with option MTM installed. Without option MTM, you must define your masks. When waveforms fall inside or outside the mask limits, the instrument can generate hard copy, save the waveform, emit a beep or SRQ, stop and wait for your input, send an E-mail message with specified data, log the time/date of the acquired waveform, or a combination of these actions.

A mask is a set of polygonal regions on the screen. Every vertical line of the screen intersects the polygon in zero, one, or two places, but never in more than two places. (A vertical line that intersects a vertical mask border is counted.) You have to break up more complicated polygons into two separate segments. Unlike limit testing, the inside of a mask is the region where waveform data would not normally fall. A telecommunications standard requires up to eight of these segments. Pulse standards always have two segments. Standards with eye patterns usually have three masks, but some have four.

You use the commands in the Mask Command Group to do the following:

- Specify the waveform source to test and the mask to use.
- Specify whether to use, and the size of, mask margins, which allow you to shrink or expand an existing set of polygons by a specified percentage.
- Specify whether to display a readout of hits and the mask on screen. You can also autofit the incoming waveforms to match the mask you choose.
- Select industry-standard masks that support a variety of electrical and optical communication standards (available with an option).
- Define and edit your own custom mask; create a new mask, or edit a standard mask (optional) to meet your needs.
- Enable, disable, or reset the mask counts. Once you turn on mask counting, it remains on until you explicitly turn it off.

Options also provide communication trigger commands that you can use with the mask commands, such as `TRIGger:A:TYPe` Communication. Refer to the Trigger Command Group.

Table 2-32: Mask commands

Command	Description
MASK?	Returns the states of all settable mask parameters
MASK:AUTOAdjust	Optimizes signal position within the mask to minimize hits
MASK:AUTOAdjust:HDELTA	Sets or queries how far autofit searches horizontally
MASK:AUTOAdjust:VDELTA	Sets or queries how far autofit searches vertically
MASK:AUTOSet:AUTOAdjust	Controls what happens at the end of a mask autoset. If ON, the horz/vert auto adjustment is run ONCE to potentially improve the result of mask autoset

Table 2-32: Mask commands (cont.)

Command	Description
MASK:AUTOSet:HPOS	Controls whether the algorithm will attempt to change the horizontal position of the signal while attempting to autoset
MASK:AUTOSet:HSCAle	Controls whether the algorithm will attempt to change the horizontal scale while attempting to autoset
MASK:AUTOSet:MODe	Controls whether an autoset will be automatically done after a standard mask is selected
MASK:AUTOSet:OFFSETAdj	Sets autoset not to enforce the rule that, for pulse standards, requires 0 V to be in a certain place in the mask. Instead, autoset will attempt to measure the DC offset in the signal and use scope offset or position controls to compensate for the DC offset in the signal
MASK:AUTOSet:STANdard	Selects the standard mask in autoset
MASK:AUTOSet:TRIGger	Controls if a `MASK:AUTOSet` affects the trigger level
MASK:AUTOSet:USER:ONE	Sets or queries the level 1 used by autoset for user masks
MASK:AUTOSet:USER:TYPe	Sets or queries how autoset scales a mask
MASK:AUTOSet:USER:ZERo	Sets or queries the level 0 used by autoset for user masks
MASK:AUTOSet:VPOS	Controls whether the autoset algorithm will affect the vertical position (or offset) of the signal while attempting to autoset
MASK:AUTOSet:VSCAle	Controls whether the autoset algorithm will attempt to change the vertical scale while attempting to autoset
MASK:COPy:USER	Copies the currently displayed mask to the User mask
MASK:COUNt	Resets to zero the number of hits for all mask segments on the specified source
MASK:COUNt:FAILURES?	Returns the number of pass/fail tests that have failed
MASK:COUNt:HITS?	Returns the total number of mask hits on all mask segments
MASK:COUNt:SEG<m>:HITS?	Returns the number of hits in mask segment <m>
MASK:COUNt:STATE	Sets or queries the mask hits count state
MASK:COUNt:TESTS?	Returns the number of pass/fail tests that have occurred
MASK:COUNt:TOTal?	Returns the sum of all hits in all mask segments
MASK:COUNt:VIOLATIONS?	Returns the number of test violations that have occurred in the current pass/fail test
MASK:COUNt:WAVEFORMS?	Returns the number of waveforms that have been acquired and processed during pass/fail testing
MASK:DISplay	Sets or queries whether the defined masks are displayed on the screen
MASK:FILTer	Sets or queries whether the selected optical reference receiver mask filter is enabled
MASK:HIGHLIGHTHits	Sets or queries whether the hits in a mask are highlighted in different colors than other waveform data
MASK:INVert	Controls whether the mask is drawn inverted
MASK:LOCk	Sets or queries the mask lock to waveform mode
MASK:MARgin:PERCent	Sets or queries the mask margin percentage
MASK:MARgin:STATE	Sets or queries the state of the mask margins
MASK:MASKPRE:AMPlitude	Sets or queries the current mask's nominal pulse amplitude in volts

Table 2-32: Mask commands (cont.)

Command	Description
MASK:MASKPRE:HSCAle	Sets or queries the nominal timing resolution used to draw the current mask in time/division
MASK:MASKPRE:HTRIGPOS	Sets or queries the nominal trigger position (pulse leading edge) used to draw the current mask as a fraction of the display width
MASK:MASKPRE:PATTERNBITS	Sets or queries the number of bits used for serial trigger for the current mask standard
MASK:MASKPRE:PRESAMPBITS	Sets or queries the number of bits before the (isolated one) pulse leading edge in the serial trigger pass/fail testing
MASK:MASKPRE:RECOrdlength	Sets or queries the nominal record length for pulse mask testing
MASK:MASKPRE:TRIGTOSAMP	Sets or queries the time in seconds, from the (leading edge) trigger position to the pulse bit sampling position
MASK:MASKPRE:VOFFSet	Sets or queries the nominal vertical offset in volts, used to vertically offset the input channels
MASK:MASKPRE:VPOS	Sets or queries the nominal vertical position in divisions, used to vertically position the input channels
MASK:MASKPRE:VSCAle	Sets or queries the nominal vertical scale in volts per division, used to vertically scale the input channels
MASK:MASKPRE:WIDth	Sets or queries the nominal bit width in seconds
MASK:POLarity	Sets or queries the input waveform polarity for the pass/fail test
MASK:SEG<m>	Deletes the specified mask segment from the Current mask
MASK:SEG<m>:NR_Pt?	Returns the number of points that make up the specified mask segment of the Current mask
MASK:SEG<m>:POINTS	Sets or queries the X-Y user coordinates of all points in the Current mask segment
MASK:SOUrce	Sets or reports which source will be compared against the mask(s) when counting is turned on
MASK:STANdard	Deletes the existing mask and sets the selected standard mask
MASK:STOPOnviolation	Sets or queries the stop on violation setting
MASK:TESt:AUX:COMPLetion	Sets or queries the `test:aux` on pass/fail test completion mode
MASK:TESt:AUX:FAILure	Sets or queries the `test:aux` status on pass/fail test failure mode
MASK:TESt:BEEP:COMPLetion	Sets or queries the beep on pass/fail test completion mode
MASK:TESt:BEEP:FAILure	Sets or queries the beep status on pass/fail test failure mode
MASK:TESt:DELay	Sets or queries the amount of time in seconds, to wait after starting pass/fail testing to evaluate waveforms
MASK:TESt:HARDCopy	Sets or queries the hard copy in pass/fail failure mode
MASK:TESt:LOG:FAILure	Sets or queries the log status on pass/fail test failure mode
MASK:TESt:REPeat	Sets or queries the repeat pass/fail test on completion mode
MASK:TESt:SAMple	Sets or queries the minimum number of waveform database (WfmDB) points the instrument can acquire before it stops a single sequence acquisition or stops running a mask test

Table 2-32: Mask commands (cont.)

Command	Description
MASK:TESt:SAMple:THReshold	Sets or queries the minimum number of hits in mask regions needed to cause the pass/fail status to change from PASSING to FAILING
MASK:TESt:SAVEWFM	Sets or queries the save waveform to file on failure setting
MASK:TESt:SAVEWFM:FILEName	Sets or queries the name of the directory and file to use with the `MASK:TESt:SAVEWFM` command
MASK:TESt:SRQ:COMPLetion	Sets or queries an SRQ on pass/fail test completion mode
MASK:TESt:SRQ:FAILure	Sets or queries the SRQ status on pass/fail test failure mode
MASK:TESt:STATE	Sets or queries the state of mask pass/fail testing
MASK:TESt:STATUS?	Returns the pass/fail test status
MASK:TESt:STOP:FAILure	Sets or queries the stop status on pass/fail test failure mode
MASK:TESt:THReshold	Sets or queries the number of failed tested waveforms needed in each pass/fail test to cause the pass/fail status to change from PASSING to FAILING
MASK:TESt:WAVEform	Sets or queries the number of waveforms to test during pass/fail testing
MASK:USER:AMPlitude	Sets or queries the current mask's nominal pulse amplitude in volts
MASK:USER:BITRate	Sets or queries the user mask's bit rate
MASK:USER:HSCAle	Sets or queries the nominal timing resolution used to draw the mask in time/division
MASK:USER:HTRIGPOS	Sets or queries the nominal trigger position (pulse leading edge) used to draw the mask as a fraction of the display width
MASK:USER:LABel	Sets or queries the label assigned to a user mask
MASK:USER:PATTERNBITS	Sets or queries the number of bits used for serial trigger for the User mask standard
MASK:USER:PRESAMPBITS	Sets or queries the number of bits before the (isolated one) pulse leading edge in the serial trigger pass/fail testing
MASK:USER:RECOrdlength	Sets or queries the nominal record length for pulse mask testing
MASK:USER:SEG<m>	Deletes the specified mask segment from the User mask
MASK:USER:SEG<m>:NR_Pt?	Returns the number of points that make up the specified User mask segment
MASK:USER:SEG<m>:POINTS	Sets or queries the X-Y user coordinates of all points in the specified User mask segment
MASK:USER:TRIGTOSAMP	Sets or queries the time in seconds, from the (leading edge) trigger position to the pulse bit sampling position
MASK:USER:VOFFSet	Sets or queries the nominal vertical offset in volts, used to vertically offset the input channels
MASK:USER:VPOS	Sets or queries the nominal vertical position in divisions, used to vertically position the input channels
MASK:USER:VSCAle	Sets or queries the nominal vertical scale in volts per division, used to vertically scale the input channels
MASK:USER:WIDth	Sets or queries the nominal bit width in seconds
RECAll:MASK	Recalls a stored mask as the User mask
SAVe:MASK	Saves the User mask to a mass storage file

Math command group

Use the commands in the Math Command Group to create and define math waveforms. You can define and display up to four math waveforms simultaneously on four channel models and up to two on two channel models. 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 waveforms you previously defined will be affected.

Math expressions can be simple, containing no mathematical computation, such as CH1, which specifies that a waveform shows the signal source of channel 1. Math expressions can also be complex, consisting of 100 plus characters and comprising many sources, functions, and operands.

The acquisition of a live waveform can stop for several reasons: You can turn off the channel, stop the waveform (via Run/Stop from the Horiz/Acq menu), or stop the trigger (via Run/Stop from the Trig menu). 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. Also, sources must exist but do not need to be displayed to be used in and to update math waveforms.

Table 2-33: Math commands

Command	Description
MATH<x>?	Returns the specified math waveform settings
MATH<x>:DEFine	Sets or queries the math<x> waveform definition for the specified waveform
MATH<x>:FILTer:MODe	Sets or queries the filter rise time parameter (centered or shifted)
MATH<x>:FILTer:RISetime	Sets or queries the filter rise time parameter
MATH<x>:LABel:NAMe	Sets or queries the label string used for annotating the displayed math waveform
MATH<x>:LABel:XPOS	Sets or queries the X screen offset where the math waveform label is displayed
MATH<x>:LABel:YPOS	Sets or queries the Y screen offset where the math waveform label is displayed
MATH<x>:NUMAVg	Sets or queries the acquisition number at which the averaging algorithm will begin exponential averaging
MATH<x>:SPECTral?	Returns the specified math waveform spectral setups
MATH<x>:SPECTral:CENTER	Sets or queries the center frequency of the spectral analyzer output data span
MATH<x>:SPECTral:GATEPOS	Sets or queries the position of the center of the gate
MATH<x>:SPECTral:GATEWIDTH	Sets or queries the gate width input to the spectral analyzer
MATH<x>:SPECTral:LOCk	Sets or queries the state of the spectral locking for the specified math waveform
MATH<x>:SPECTral:MAG	Sets or queries the units of the SpectralMag function in the specified math string
MATH<x>:SPECTral:PHASE	Sets or queries the units of the SpectralPhase function in the specified math string

Table 2-33: Math commands (cont.)

Command	Description
MATH<x>:SPECTral:REFLevel	Sets or queries the vertical position of the output data from the spectral analyzer
MATH<x>:SPECTral:REFLEVELOffset	Sets or queries the spectral reference level offset used for calculating the dB value
MATH<x>:SPECTral:RESBw	Sets or queries the resolution bandwidth of the spectral analyzer
MATH<x>:SPECTral:SPAN	Sets the top of the frequency span to a value that is closest to the specified value Or queries the specified math waveform's current frequency span value
MATH<x>:SPECTral:SUPPress	Sets or queries the phase suppression threshold
MATH<x>:SPECTral:UNWRap	Enables or disables or returns the state of the phase unwrap of the spectral analyzer output data
MATH<x>:SPECTral:WINdow	Sets or queries the window function used to multiply the input data to the spectral analyzer
MATH<x>:THRESHold	Sets or queries the comparable threshold for converting the math signal to digital form
MATH<x>:UNITString	Sets or queries the units used for the math waveform; command overrides the default unit string
MATH<x>:VERTical:AUTOSCale	Sets or queries auto-scaling of the specified math waveform
MATH<x>:VERTical:POSition	Sets or queries the specified math waveform vertical position
MATH<x>:VERTical:SCAle	Sets or queries the specified math waveform vertical scale (per div)
MATHArbflt<x>:FILepath	Sets the file path for a file of filter coefficients and reads the file
MATHArbflt<x>:READFile	Forces a reread of the filter file for the specified arbitrary filter
MATHVAR?	Sets or queries all 8 numerical values you can use within math expressions
MATHVAR:VAR<x>	Sets or queries a numerical value you can use within math expressions

Measurement command group

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

Up to eight automated measurements can be displayed on the screen. In the commands, these measurement readouts are named MEAS<x>, where <x> can be 1 through 8.

In addition to the eight 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 parameters, such as waveform sources and reference levels, differently for each measurement.

Table 2-34: Measurement commands

Command	Description
MEASUrement?	Returns all measurement parameters
MEASUrement:ANNOTation:IMMEDSTAte	Sets or queries the state of immediate measurement annotation
MEASUrement:ANNOTation:NUMX?	Returns the number of horizontal measurement annotations being displayed
MEASUrement:ANNOTation:NUMY?	Returns the number of vertical measurement annotations being displayed
MEASUrement:ANNOTation:STATE	Sets or queries the state of visible measurement annotations
MEASUrement:ANNOTation:TYPe	Sets or queries the type of annotations being used
MEASUrement:ANNOTation:X<x>?	Returns the value of the specified annotation
MEASUrement:ANNOTation:Y<x>?	Returns the value of the specified annotation
MEASUrement:GATing	Sets or queries the measurement gating
MEASUrement:IMMed?	Returns all immediate measurement setup parameters
MEASUrement:IMMed:DELay?	Returns information about the immediate delay measurement
MEASUrement:IMMed:DELay:DIREction	Sets or queries the search direction to use for immediate delay measurements
MEASUrement:IMMed:DELay:EDGE[1]	Sets or queries the slope of the edge used for immediate delay “from” waveform measurements
MEASUrement:IMMed:DELay:EDGE2	Sets or queries the slope of the edge used for immediate delay “to” waveform measurements
MEASUrement:IMMed:METHod	Specifies or queries the method used to calculate the 0% and 100% reference level for immediate measurements
MEASUrement:IMMed:NOISe	Sets or queries the location of the noise measurement on the eye waveform
MEASUrement:IMMed:REFLevel?	Returns the reference levels

Table 2-34: Measurement commands (cont.)

Command	Description
MEASUrement:IMMed:REFLevel:ABSolute:HIGH	Sets or queries the high reference level
MEASUrement:IMMed:REFLevel:ABSolute:LOW	Sets or queries the low reference level
MEASUrement:IMMed:REFLevel:ABSolute:MID<x>	Sets or queries the middle reference levels
MEASUrement:IMMed:REFLevel:METHod	Sets or queries the method used to calculate measurements
MEASUrement:IMMed:REFLevel:PERCent:HIGH	Sets or queries the top level for rise time (90% level)
MEASUrement:IMMed:REFLevel:PERCent:LOW	Sets or queries the low level for rise time (10% level)
MEASUrement:IMMed:REFLevel:PERCent:MID<x>	Sets or queries the mid levels for measurements
MEASUrement:IMMed:SOUrce<x>	Sets or queries the source to measure “to” or “from” for phase or delay immediate measurements
MEASUrement:IMMed:SOUrce<x>:SIGType	Sets or queries the type of input signal used for the specified measurement source
MEASUrement:IMMed:TYPe	Sets or queries 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:MEAS<x>?	Returns all measurement parameters for the displayed measurement
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 specified measurement
MEASUrement:MEAS<x>:DELay:DIREction	Sets or queries the search direction to use for delay measurements
MEASUrement:MEAS<x>:DELay:EDGE<x>	Sets or queries the slope of the edge to use for delay “from” or “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>:METHod	Sets or queries the method used to calculate the 0% and 100% reference level
MEASUrement:MEAS<x>:MINImum?	Returns the minimum value found since the last statistical reset
MEASUrement:MEAS<x>:NOISe	Sets or queries the location of the noise measurement on the eye waveform for MEAS<x>
MEASUrement:MEAS<x>:REFLevel?	Returns the reference levels
MEASUrement:MEAS<x>:REFLevel:ABSolute:HIGH	Sets or queries the top level for rise time (90% level)
MEASUrement:MEAS<x>:REFLevel:ABSolute:LOW	Sets or queries the low level for rise time (10%)
MEASUrement:MEAS<x>:REFLevel:ABSolute:MID<x>	Sets or queries the mid1 or mid2 level for measurements
MEASUrement:MEAS<x>:REFLevel:METHod	Sets or queries the reference level units used for measurement calculations

Table 2-34: Measurement commands (cont.)

Command	Description
MEASUrement:MEAS<x>:REFLevel:PERCent:HIGH	Sets or queries the top level for rise time (90% level)
MEASUrement:MEAS<x>:REFLevel:PERCent:LOW	Sets or queries the low level for rise time (10% level)
MEASUrement:MEAS<x>:REFLevel:PERCent:MID<x>	Sets or queries the mid1 or mid2 level for delay measurements
MEASUrement:MEAS<x>:SOUrce<x>	Sets or queries the source to measure “from” or “to” for phase or delay measurements
MEASUrement:MEAS<x>:SOUrce<x>:SIGType	Sets or queries the type of input signal used for the specified measurement source for the specified measurement slot
MEASUrement:MEAS<x>:STATE	Sets or queries 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 queries 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 queries the method used for calculating reference levels
MEASUrement:NOISe	Sets or queries the location of the noise measurement on the eye waveform
MEASUrement:REFLevel:ABSolute:HIGH	Sets or queries the top reference level for rise time
MEASUrement:REFLevel:ABSolute:LOW	Sets or queries the low reference level for rise time
MEASUrement:REFLevel:ABSolute:MID<x>	Sets or queries the mid1 or mid2 reference level for measurements
MEASUrement:REFLevel:METHod	Sets or queries the method for assigning high and low reference levels
MEASUrement:REFLevel:PERCent:HIGH	Sets or queries the top reference percent level for rise time
MEASUrement:REFLevel:PERCent:LOW	Sets or queries the low reference percent level for rise time
MEASUrement:REFLevel:PERCent:MID<x>	Sets or queries the mid1 or mid 2 reference percent level for waveform measurements
MEASUrement:SOUrce<x>:SIGType	Sets or queries the type of input signal used for the specified measurement source
MEASUrement:STATIstics:COUNt	Clears existing measurement statistics from memory
MEASUrement:STATIstics:MODe	Turns measurement statistics on or off and sets or queries which pair of statistical accumulations is displayed
MEASUrement:STATIstics:WEIghting	Sets or queries the 'time constant' for mean and standard deviation statistical accumulations

Miscellaneous command group

Miscellaneous commands do not fit into other categories.

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

Table 2-35: Miscellaneous commands

Command	Description
APPLication:ACTivate	Starts the application specified in the string
APPLication:SCOPEAPP:WINDOW	Sets or queries whether the oscilloscope application is minimized or displayed full screen
AUXout?	Returns the auxiliary out setup
AUXout:EDGE	Sets or queries the direction in which the trigger output signal will transition when a trigger occurs
AUXout:SOUrce	Sets or queries the trigger source at the BNC connection
AUTOSet	Sets the vertical, horizontal, and trigger controls of the instrument to automatically acquire and display the selected waveform
BELl	In previous models, this command sounded the audio indicator. Provided for backward compatibility
CHANNELMAPping?	Returns a list of all channel mappings
CLEAR	Clears acquisitions, measurements, and waveforms
CMDBatch	Turns command batching on or off Or returns the state of command batching
DATE	Sets or queries the date that the instrument can display
*DDT	Sets or queries the commands that will be executed by the group execute trigger
FPANel:PRESS	Turns off the displayed menu
HEADer	Sets or queries the Response Header Enable State
HDR	Sets or queries the Response Header Enable State
ID?	Returns identifying information about the instrument and its firmware
*IDN?	Returns the instrument identification code
IDNMultiscope?	Returns instrument data on all MultiScope instruments.
IDNMultiscope:A?	Returns the *IDN? response for the Master instrument.
IDNMultiscope:B?	Returns the *IDN? response for Extension 1.
IDNMultiscope:C?	Returns the *IDN? response for Extension 2.
IDNMultiscope:D?	Returns the *IDN? response for Extension 3.
LOCk	Sets or queries the front panel lock state
*LRN?	Returns a listing of instrument settings
MULTiscope:CONFig	Sets or queries the current MultiScope operational configuration.
MULTiscope:EXIT	Instructs the Configuration Managers (and active scope applications) across the entire MultiScope configuration to shutdown.

Table 2-35: Miscellaneous commands (cont.)

Command	Description
MULTiscope:RESTART	Instructs the Configuration Managers (and active scope applications) across the entire MultiScope configuration to exit and restart themselves.
MULTiscope:STATUS?	Returns the current state of the MultiScope system.
NEWpass	Changes the password for user protected data
PASSWord	Provides access for changing user protected data
REM	Specifies a comment which is ignored by the instrument
ROSc:OUT:FREQuency	Sets or queries the selected frequency for the timebase reference output signal
ROSc:OUT:ULTRAsync	Sets or queries the state of the UltraSync 12.5 GHz Clock Out.
ROSc:SOUrce	Sets or queries the selected source for the time base reference oscillator
ROSc:STATE?	Returns whether the time base reference oscillator is locked
ROSc:TRACking	Sets or queries the selected source for the time base reference oscillator. The reference oscillator locks to this source
SET?	Returns a listing of instrument settings
TEKSecure	Initializes both waveform and setup memories
TIME	Sets or queries the time displayed by the instrument
*TRG	Performs the group execute trigger (GET)
*TST?	Tests the GPIB interface and returns status
UNLock	Unlocks front panel
USBTMC:PRODUCTID:DECimal?	Queries the USBTMC product ID in decimal
USBTMC:PRODUCTID:HEXadecimal?	Queries the USBTMC product ID in hexadecimal
USBTMC:SERIALnumber?	Queries the instrument serial number
USBTMC:VENDORID:DECimal?	Queries the USBTMC product ID in decimal
USBTMC:VENDORID:HEXadecimal?	Queries the USBTMC product ID in hexadecimal
VERBose	Sets or queries the verbose state

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 instrument. When you recall a setup, the instrument restores itself to the state that it was in when you originally saved that setting.

Table 2-36: Save and Recall commands

Command	Description
*RCL	Restores the state of the instrument from a copy of the setting stored in memory
*SAV	Stores the current instrument state to a specified memory location
*SDS	Changes the specified setup to reference the factory setup instead of the user setup
DELEte:SETUp	Removes stored setups from memory and initializes the location with the factory default setups
DELEte:WAVEform	Deletes (one or all of) the stored reference waveforms from memory
FACtory	Resets the instrument to factory default settings
RECAll:SETUp	Recalls saved instrument settings
RECAll:SETUp:DESKew	Sets or queries the deskew values that are affected by a default setup or a recalled setup
RECAll:WAVEform	Recalls a stored waveform to a reference location
SAVe:EVENTtable:BUS<x>	Saves the event table contents of the specified bus in CSV format, to the file that you specify by <string>
SAVe:MARKS	Saves the user search marks in CSV format, to the file that you specify
SAVe:SETUp	Saves the current front panel setup to a specified memory location or file
SAVe:WAVEform	Saves a waveform to one of four reference memory locations or a mass storage file
SAVe:WAVEform:DATa:STARt	Sets or queries save waveform starting waveform data sample.
SAVe:WAVEform:DATa:STOP	Sets or queries save waveform ending waveform data sample.
SAVe:WAVEform:FILEFormat	Sets or queries the format for saved waveforms
SAVe:WAVEform:FORCESAMEFilesize	Sets or queries the save waveform force same file size feature
SETUp:NAMe	Sets or queries the user-defined setup label

Save On command Group

Use this group of commands to program the oscilloscope to save images, measurements, waveforms, or the instrument setup, on triggers that you select.

Table 2-37: Save On commands

Command	Description
SAVEON	The Reset argument sets the auto-increment file count to 0
SAVEON:COUNt?	Returns the number of events (files) that have been saved since the last reset
SAVEON:FILE:AUTOInc	Sets or queries the state of the auto increment file name feature (on or off)
SAVEON:FILE:COUNt	Sets or queries the starting number that will be appended to the custom file name when SAVEON:FILE:AUTOInc is enabled and an image, measurement, or waveform is saved
SAVEON:FILE:DEST	Sets or queries the location where files are saved
SAVEON:FILE:NAME	Sets or queries the file name to use when the file type is set to Custom
SAVEON:FILE:TYPE	Sets or queries whether to use the data and time as the file name (auto) or to use a custom file name
SAVEON:IMAGe	Sets or queries whether to save a screen capture when a limit test failure, mask failure, or trigger occurs
SAVEON:LIMit	Sets or queries whether to save a file when there is a limit failure
SAVEON:MASK	Sets or queries whether to save a file when there is a mask failure
SAVEON:MEASUrement	Sets or queries whether to save a measurement when a limit test failure, mask failure, or trigger occurs
SAVEON:NUMEvents	Sets or queries the maximum number of events that will be saved
SAVEON:SETUP	Saves the instrument setup when there is an event. Queries the status of this feature (on or off)
SAVEON:TRIGger	Sets or queries whether to save a file when a trigger occurs
SAVEON:WAVEform	Sets or queries whether to save a waveform when a limit test failure, mask failure, or trigger occurs

Search and Mark command group

Use search and mark commands to seek out and identify information in waveform records that warrant further investigation.

Table 2-38: Search and Mark commands

Command	Description
MARK	Moves to the next or previous reference mark, or queries the current mark setup
MARK:CREATE	Creates a mark on a specified waveform or all waveforms in a column
MARK:DELEte	Deletes a mark on a particular waveform, all waveforms in a column, the selected mark, or all marks
MARK:FREE?	Returns how many user marks are available for use
MARK:SELECTED:END?	Returns the end of the selected mark, 0 to 100% of the waveform
MARK:SELECTED:FOCUS?	Returns the focus of the selected mark, 0 to 100% of the waveform
MARK:SELECTED:LABel	Sets or queries the label for the selected mark
MARK:SELECTED:MARKSINCOLumn?	Returns the number of marks in the current zoom pixel column
MARK:SELECTED:OWNer?	Returns the owner of the selected mark
MARK:SELECTED:SOUrce?	Sets or queries the source waveform for the selected mark
MARK:SELECTED:STARt?	Returns the starting point of the selected mark, 0 to 100% of the waveform
MARK:SELECTED:STATE	Returns the on or off state of the selected mark. The selected mark is at or near the center of the display
MARK:SELECTED:ZOOm:POSition?	Returns the position of the selected mark, 0 to 100% of the zoom overview window
MARK:SELECTED:ZOOm:SCAle?	Returns the scale of the selected mark, 0 to 100% of the zoom overview window
MARK:TOTal?	Returns how many marks are currently in use
SEARCH:MARKALLevents	Sets or queries the mark all trigger events setting
SEARCH:SEARCH<x>	Converts the specified search marks to the user marks
SEARCH:SEARCH<x>:COPy	Copies the search criteria to the trigger, the trigger criteria to a search, or copy search to search
SEARCH:SEARCH<x>:STATE	Selects which search is selected
SEARCH:SEARCH<x>:TOTAL?	Returns the total number of matches for the search. The total number of matches may be more than the number of marks placed
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOBitrate	Sets or queries the auto bit rate setting to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:CONDition	Sets or queries the CAN bus trigger condition
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:VALue	Sets or queries the CAN bus trigger data value
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:FRAMEtype	Sets or queries CAN bus trigger frame type
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:FORMat	Sets or queries the CAN bus trigger data format

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:SIZe	Sets or queries the CAN bus trigger data size
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:QUALifier	Sets or queries the CAN bus trigger data qualifier
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:DIRection	Sets or queries the CAN bus identifier direction
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:FORMat	Sets or queries the CAN bus trigger identifier format
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:MODe	Sets or queries the CAN bus trigger identifier mode
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:VALue	Sets or queries CAN bus trigger identifier value
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition	Specifies a field or condition within an Ethernet frame to search on
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:OFFSet	Specifies the data offset value, in bytes, to use when searching on the Ethernet data field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe	Specifies the number of contiguous TCP/IPv4/MAC client bytes to use when searching on the Ethernet data field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:VALue	Specifies the binary value to use when searching on the Ethernet data field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue	Specifies the 32-bit value to use when searching on the Ethernet IPv4 header address destination field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet IPv4 header address destination field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue	Specifies the 8-bit value to use when searching on the Ethernet IPv4 header protocol field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat	Specifies the data format (binary or hex) of the bit pattern to use when searching on the Ethernet IPv4 header protocol field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue	Specifies the 32-bit value to use when searching on the Ethernet IPv4 header address source field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat	Specifies the data format (binary or hex) of the bit pattern to use when searching on the Ethernet IPv4 header address source field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue	Specifies the 48-bit value to use when searching on the Ethernet MAC address destination field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat	Specifies the data format (binary or hex) of the bit pattern to use when searching on the Ethernet MAC address destination field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet MAC address source field

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue	Specifies the 48-bit value to use when searching on the Ethernet MAC address source field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue	Specifies the 16-bit value to use when searching on the Ethernet MAC length/type
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet MAC length or type
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:VALue	Specifies the 32-bit value to use when searching on the Ethernet Q-Tag field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:FORMat	Specifies the data format (binary or hex) to use when searching on the Q-Tag field
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier	Specifies the qualifier to be used when the Ethernet search condition is set to MACLENgth or DATa
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue	Specifies the 32-bit value to use when searching on the Ethernet TCP header acknowledgement number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header acknowledgement number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue	Specifies the 16-bit value to use when searching on the Ethernet TCP header destination port number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header destination port number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue	Specifies the 32-bit value to use when searching on the Ethernet TCP header sequence number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header sequence number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue	Specifies the 16-bit value to use when searching on the Ethernet TCP header source port number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header source port number
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet data field
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CONDition	Sets or queries FLEXRAY bus search trigger condition
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat	Sets or queries the FLEXRAY bus search cycle count format
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier	Sets or queries the FLEXRAY bus search trigger cycle count qualifier
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue	Sets or queries the FLEXRAY bus search cycle count value
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:FORMat	Sets or queries FLEXRAY bus search data format

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet	Sets or queries the FLEXRAY bus search trigger data offset
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier	Sets or queries the FLEXRAY bus search trigger data qualifier
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:SIZe	Sets or queries the FLEXRAY bus search trigger data size
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:VALue	Sets or queries the FLEXRAY bus search trigger data value
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:EOFTYPE	Sets or queries the FLEXRAY bus search trigger end of file type
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:ERRTYPE	Sets or queries the FLEXRAY bus search trigger error type
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:FRAMEType	Sets or queries the FLEXRAY bus search trigger frame type
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC	Sets or queries the FLEXRAY bus search trigger header CRC
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount	Sets or queries the FLEXRAY bus search trigger header cycle count
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID	Sets or queries the FLEXRAY bus search trigger header frame id
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits	Sets or queries the FLEXRAY bus search header Indicator Bits
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth	Sets or queries the FLEXRAY bus search trigger header payload length
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat	Sets or queries the FLEXRAY bus search trigger identifier format
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier	Sets or queries the FLEXRAY bus search trigger identifier qualifier
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue	Sets or queries the FLEXRAY bus search trigger identifier value
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:CONDition	Sets or queries the LIN bus search trigger condition
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:FORMat	Sets or queries the LIN bus search trigger data format
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:QUALifier	Sets or queries the LIN bus search trigger data qualifier
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:SIZe	Sets or queries the LIN bus search trigger data size
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:VALue	Sets or queries the LIN bus search trigger data value
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:ERRTYPE	Sets or queries the LIN bus search trigger error type

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:FORMat	Sets or queries the LIN bus search trigger identifier format
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:VALue	Sets or queries the LIN bus search trigger identifier value
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:FORMat	Sets or queries the address format for the I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:MODe	Sets or queries the address mode for the specified I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:TYPe	Sets or queries the address type for the specified I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:VALue	Sets or queries the address string when the search condition for the specified search is Address or AddressData, to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:CONDition	Sets or queries the trigger condition for the specified I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:DIRection	Sets or queries the direction of the data for the I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:FORMat	Sets or queries the trigger data format for the specified I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:SIZe	Sets or queries the length of the data string in bytes used for the specified I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:VALue	Sets or queries the data value of the data token for the specified I2C bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat	Sets the data format of the address (either binary or hex) to use in a MIL-STD-1553 bus command word search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue	Specifies the value of the 5-bit remote terminal address to use in a MIL-STD-1553 bus command word search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier	Specifies the qualifier to use when searching on a MIL-STD-1553 bus command word remote terminal address field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt	Specifies the bit pattern for the 5-bit Word Count/Mode Code sub-address field to use in a MIL-STD-1553 bus command word trigger
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat	Specifies the data format (binary or hex) of the bit pattern for the 5-bit Word Count/Mode Code field to use in a MIL-STD-1553 bus command word search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:PARity	Specifies the parity to use in a MIL-STD-1553 bus command word search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress	Specifies the 5 bit sub-address to use in a MIL-STD-1553 bus command word sub-address search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:FORMat	Specifies the data format (binary or hex) of the sub-address field bit pattern to use in a MIL-STD-1553 bus command word sub-address search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:TRBit	Specifies to search on either a “transmit” or a “receive” signal for the transmit/receive bit (bit 9), for a MIL-STD-1553 bus command word search

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition	Specifies which word type or condition within a MIL-STD-1553 bus word to search on
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:FORMat	Specifies the format (binary or hex) of the bit pattern to use for a MIL-STD-1553 bus data word search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:PARity	Specifies the parity to use when searching on a MIL-STD-1553 bus data word parity bit
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:VALue	Specifies the value to use for a MIL-STD-1553 bus data word search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:ERRTYPE	Specifies the signaling error type to use for a MIL-STD-1553 bus data word search: Parity, Sync, Manchester, or Data
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity	Specifies the parity to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:FORMat	Specifies the format (binary or hex) of the bit pattern to use when searching on the MIL-STD-1553 status address field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier	Specifies the qualifier to use when searching on the MIL-STD-1553 status address field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue	Specifies the value of the 5-bit remote terminal address to use when searching on the MIL-STD-1553 status address field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR	Specifies the status word broadcast command received (BCR) bit value (bit 15) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY	Specifies the status word busy bit value (bit 16) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA	Specifies the status word dynamic bus control acceptance (DBCA) bit value (bit 18) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR	Specifies the status word instrumentation bit value (bit 10) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME	Specifies the status word message error bit value (bit 9) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ	Specifies the status word service request (SRQ) bit value (bit 11) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF	Specifies the status word subsystem flag bit value (bit 17) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF	Specifies the status word terminal flag bit value (bit 19) to use when searching on the MIL-STD-1553 status bit field
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit	Specifies either the minimum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit, or else it specifies the minimum inter-message gap (IMG)
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit	Specifies either the maximum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit, or it specifies the maximum inter-message gap (IMG)
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:QUALifier	Specifies the qualifier to use when triggering on either the remote terminal response time (RT) or the inter-message gap (IMG)

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:CONDition	Sets or queries the search condition for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTLong	Sets or queries the data type for long packets for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTShort	Sets or queries the data type for short packets for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ERRor	Sets or queries the error type for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ESCape	Sets or queries the Escape command for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:OFFset:VALue	Sets or queries the offset value for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:FORMat	Sets or queries the format of the packet data for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:VALue	Sets or queries the value of the packet data string for a specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat	Sets or queries the generic payload format for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe	Sets or queries the generic payload size of the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:VALue	Sets or queries the generic payload value string of the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PIXel:VALue	Sets or queries the pixel value for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:QUAlifier	Sets or queries the RAW qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:VALue	Sets or queries the RAW value for the specified MIPI CSI2 bus trigger search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:QUAlifier	Sets or queries the RGB Blue qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:VALue	Sets or queries the RGB blue value for the for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:QUAlifier	Sets or queries the RGB blue value for the for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:VALue	Sets or queries the RGB Green value for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:QUAlifier	Sets or queries the RGB Red qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:VALue	Sets or queries the RGB Red value for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:VC	Sets or queries the Virtual Channel for the specified MIPI CSI2 bus trigger search to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WARNing	Sets or queries the warning type for a specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WC:VALue	Sets or queries the word count value for a specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:QUAlifier	Sets or queries the YUV U qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:VALue	Sets or queries the YUV U value for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:QUAlifier	Sets or queries the YUV V qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:VALue	Sets or queries the YUV V value for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:QUAlifier	Sets or queries the YUV Y qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:VALue	Sets or queries the YUV Y value for the specified MIPI CSI2 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ACKERRreport	Sets or queries the Acknowledge and Error Report for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:CONDition	Sets or queries the search condition for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DIRection	Sets or queries the direction for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTLong	Sets or queries the data type for long packets for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTShort	Sets or queries the data type for short packets for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ERRor	Sets or queries the error type for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ESCape	Sets or queries the Escape Command for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue	Sets or queries the offset value for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:FORMat	Sets or queries the format of the packet data for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUEOne	Sets or queries the value of the first byte (8 bits) of a packet data string for a specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUETwo	Sets or queries the value of the second byte (8 bits) of a packet data string for a specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:FORMat	Sets or queries the generic payload format for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:SIZe	Sets or queries the generic payload size of the specified MIPI DSI1 bus trigger search to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:VALue	Sets or queries the generic payload value string of the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PIXel:VALue	Sets or queries the pixel value for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:QUAlifier	Sets or queries the RGB Blue qualifier for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:VALue	Sets or queries the RGB blue value for the for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:QUAlifier	Sets or queries the RGB Green qualifier for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:VALue	Sets or queries the RGB Green value for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:QUAlifier	Sets or queries the RGB Red qualifier for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:VALue	Sets or queries the RGB Red value for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:VC	Sets or queries the Virtual Channel for the specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WARNing	Sets or queries the warning type for a specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WC:VALue	Sets or queries the word count value for a specified MIPI DSI1 bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:QUAlifier	Sets or queries MIPI DSI-1 CB qualifier for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue	Sets or queries MIPI DSI-1 CB value for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:QUAlifier	Sets or queries MIPI DSI-1 CR qualifier for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue	Sets or queries MIPI DSI-1 CR value for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier	Sets or queries MIPI DSI-1 Y qualifier for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue	Sets or queries MIPI DSI-1 Y value for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:FORMat	Sets or queries the data format for a parallel bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:VALue	Sets or queries the data value for a parallel bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR	Sets or queries the PCIE bus search trigger character
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus	Sets or queries the PCIE bus search character symbol minus

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS	Sets or queries the PCIE bus search trigger character symbol plus
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CONDition	Sets or queries the PCIE bus search trigger condition
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:DISParity	Sets or queries the PCIE bus search trigger disparity
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:CHAR	Sets or queries the PCIE bus search trigger error character
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:DISParity	Sets or queries the PCIE search trigger error disparity
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:ORDERedset	Sets or queries the PCIE bus search trigger error ordered set
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:SYNC	Sets or queries the triggering on SYNC error state.
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:FORMat	Sets or queries the PCIE search trigger format
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:LONGORDERedset	Sets or queries the PCIE bus search trigger long ordered set
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>	Sets or queries the PCIE bus search pattern character
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset	Sets or queries the PCIE bus search trigger pattern ordered set
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>	Sets or queries the PCIE bus search trigger pattern symbol minus
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>	Sets or queries the PCIE bus search pattern symbol plus
SEARCH:SEARCH<x>:TRIGer:A:BUS:PCIE:SYNCHeader	Sets or queries the PCIE search sync header.
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:CONDition	Sets or queries the search condition for the specified RS232c bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:FORMat	Sets or queries the data format for the specified RS232c bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:SIZe	Sets or queries the number of bits per word in the data string for the specified RS232c bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:VALue	Sets or queries the data string value for the specified RS232c bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:CHAR	Sets or queries the single 8B10B character in the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus	Sets or queries the specified 8B10B character (negative disparity) for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS	Sets or queries the specified 8B10B character (positive disparity) for the specified search

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition	This command sets or queries the trigger condition for the specified 8B10B bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:DISParity	Sets the disparity to positive, negative, or either
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:CHAR	Sets the character error search to on or off, or queries the status of the feature
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:DISParity	Sets the disparity error search to on or off, or queries the status of this feature
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat	Sets the format for the specified search to character or symbol, or queries the state of this feature
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>	Sets or queries the first, second, third or fourth character in the pattern for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>	Sets or queries the specified 8B10B character (negative disparity) for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>	Sets or queries the specified 8B10B character (positive disparity) for the specified search
SEARCH:SEARCH<x>:TRIGger:A:BUS:SOUrce	Sets or queries the bus source for the specified bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:CONDition	Sets or queries the trigger condition for the specified SPI bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:FORMat	Sets or queries the data format for the specified SPI bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:SIZe	Sets or queries the length of the data string in bytes used for the specified SPI bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:VALue	Sets or queries the data value of the data token for the specified SPI bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:FORMat	Sets or queries the address format for normal token for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:HIVALue	Sets or queries the address value for normal token to be used with In Range and Out of Range qualifiers for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:VALue	Sets or queries the address value for normal token for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:CHAR	Sets or queries the single USB character in the specified search.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus	Sets or queries the specified USB character bit pattern (negative disparity) for the specified search.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS	Sets or queries the specified character bit pattern (positive disparity) for the specified search.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CONDition	Sets or queries the search condition for the specified USB bus trigger search to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:FORMat	Sets or queries the data format for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:HIVALue	Sets or queries the data value for data token used with In Range and Out of Range qualifiers for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:OFFSet	Sets or queries data offset for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:SIZe	Sets or queries the length of the data string in bytes for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:TYPe	Sets or queries the data packet type for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:VALue	Sets or queries the data value for data token for the specified bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DISParity	Sets or queries the disparity.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:FORMat	Sets or queries the endpoint format for data token (to be used with In Range and Out of Range qualifiers) for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:HIVALue	Sets or queries the endpoint value for data token used with In Range and Out of Range qualifiers for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:VALue	Sets or queries the endpoint value for normal token for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:CHAR	Sets the character error search to on or off, or queries the status of the feature.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:DISParity	Sets or queries the disparity error search (on or off).
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:ORDERedset	Sets or queries the state of searching for an ordered set in the specified search.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERRTYPE	Sets or queries the error type for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:FORMat	Sets the format for the specified search to character or symbol, or queries the state of this feature.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:HANDSHAKEType	Sets or queries the handshake type for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:LONGORDERedset	Sets or queries the USB bus search trigger long ordered set.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets	Sets or queries the USB bus search packets to mark.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:LMPPACKets	Sets or queries the LMP packets to mark when system is searching for LMP packets.

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:TPPACKets	Sets or queries the TP packets to mark when the system is searching for TP packets.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:CHAR<x>	Sets or queries the first, second, third or fourth character in the pattern for the specified search.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:NUMSymbols	Sets or queries the number of symbols to search for when USB search type is in pattern mode.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:ORDERedset	Sets or queries the state of searching for an ordered set.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>	Sets or queries the specified character bit pattern (negative disparity) for the specified search.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>	Sets or queries the specified character bit pattern (positive disparity) for the specified search.
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:QUALifier	Sets or queries the qualifier for address and data for the specified bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FORMat	Sets or queries the format for the Start of Frame frame number for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FRAMENUMber	Sets or queries the frame number string to use for the Start of Frame for the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPECIALType	Sets or queries the PID value for the USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:ET:VALue	Sets or queries the Endpoint Type value for the specified USB bus trigger split token field search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:FORMat	Sets or queries the format for the hub address of the specified USB bus trigger on split token field search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:VALue	Sets or queries the hub address of the specified USB bus trigger on split token field search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:FORMat	Sets or queries the port address format for the specified USB bus trigger on split token field search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:VALue	Sets or queries the port address for the specified USB bus trigger on split token field search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SC:VALue	Sets or queries the Start/Complete value for the specified USB bus trigger on split token field search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SE:VALue	Sets or queries the Start/End value for the specified USB bus trigger on split token field search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:TOKENType	Sets or queries the token type when the specified USB bus trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel	Sets or queries the auto reference levels setting to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:BITRate	Sets or queries the DDR memory bit rate to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSActive	Sets or queries whether the expected chip-select will be an active HIGH or LOW to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSLevel	Sets or queries the DDR memory search chip-select level to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSMode	Sets or queries the chip-select mode to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSSource	Sets or queries the chip select source
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:DATASource	Sets or queries the DDR memory data source to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:HYSteresis	Sets or queries the hysteresis percent for DDR memory searches to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MARgin	Sets or queries the termination logic margin for back-to-back writes
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MAXCas	Sets or queries the maximum distance in clock periods, to the left of the start of the mark where the instrument looks for the chip-select pulse
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MINCas	Sets or queries the minimum distance in clock periods, to the left of the start of the mark where the instrument looks for the chip-select pulse
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:HIGH	Sets or queries the value of the data HIGH reference level to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:LOW	Sets or queries the value of the data LOW reference level to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:MID	Sets or queries the value of the data MID reference level to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:HIGH	Sets or queries the value of the strobe HIGH reference level to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:LOW	Sets or queries the value of the strobe LOW reference level to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:MID	Sets or queries the value of the strobe MID reference level
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STANdard	Sets or queries the DDR standard type for a DDR Memory analysis search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STROBEsource	Sets or queries the strobe source for a DDR Memory analysis search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:CH<x>	Sets or queries the slope for an edge trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:MATH<x>	Sets or queries the slope for an edge trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:REF<x>	Sets or queries the slope for an edge trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOUrce	Sets or queries the source waveform for an edge trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:CH<x>	Sets or queries the polarity for a glitch trigger search to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:MATH<x>	Sets or queries the polarity for a glitch trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:REF<x>	Sets or queries the polarity for a glitch trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:GLItch:TRIGIF	Sets or queries the accept or reject state for a glitch trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:GLItch:WIDth	Sets or queries the glitch width for a glitch trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x>	Sets or queries the level for an edge trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LEVel:MATH<x>	Sets or queries the math waveform level for an edge trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LEVel:REF<x>	Sets or queries the specified reference waveform level for a trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:FUNCtion	Sets or queries the logic operator for a pattern or state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:CH<x>	Sets or queries the Boolean logic criteria for a pattern or state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:MATH<x>	Sets or queries the Boolean logic criteria for a pattern or state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:REF<x>	Sets or queries the Boolean logic criteria for a pattern or state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x>	Sets or queries the channel threshold level for a pattern or state trigger search to determine where to place a mark. SEARCH<x> is the search number and CH<x> is the channel number
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH<x>	Sets or queries the math waveform threshold level for a pattern or state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x>	Sets or queries the reference waveform threshold level for a pattern or state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn	Sets or queries the pattern trigger pattern test that determines where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:LESSLimit	Sets or queries the time the pattern trigger less limit test uses to determines where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:MORELimit	Sets or queries the time the pattern trigger more limit test uses to determines where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:PULse:SOUrce	Sets or queries the source waveform for a pulse trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:CH<x>	Sets or queries the polarity setting for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:MATH<x>	Sets or queries the polarity setting for a runt trigger search to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:REF<x>	Sets or queries the polarity setting for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:CH<x>	Sets or queries the channel threshold HIGH level for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:MATH<x>	Sets or queries the math waveform threshold HIGH level for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:REF<x>	Sets or queries the reference waveform threshold HIGH level for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:CH<x>	Sets or queries the channel threshold LOW level for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:MATH<x>	Sets or queries the math waveform threshold LOW level for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:REF<x>	Sets or queries the reference waveform threshold LOW level for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn	Sets or queries the condition setting for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth	Sets or queries the width setting for a runt trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:EDGE	Sets or queries the clock slope setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:SOUrce	Sets or queries the clock source setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:CH<x>	Sets or queries the clock threshold setting for a setup/hold trigger search
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:MATH<x>	Sets or queries the clock threshold setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:REF<x>	Sets or queries the clock threshold setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:SOUrce	Sets or queries the data source setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:CH<x>	Sets or queries the data threshold setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:MATH<x>	Sets or queries the data threshold setting for a setup/hold trigger search
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:REF<x>	Sets or queries the threshold high voltage level setting for a transition trigger search
SEARCH:SEARCH<x>:TRIGger:A:SETHold:HOLDTime	Sets or queries the hold time setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:SETHold:SETTime	Sets or queries the setup time setting for a setup/hold trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:EDGE	Sets or queries the clock slope setting for a state trigger search to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:SOUrce	Sets or queries the clock source of the twelve waveforms for a state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:STATE:WHEn	Sets or queries the state trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:CH<x>	Sets or queries the polarity for a timeout trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:MATH<x>	Sets or queries the polarity for a timeout trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:REF<x>	Sets or queries the polarity for a timeout trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:TIMe	Sets or queries the time setting for a timeout trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:DELTATime	Sets or queries the transition time setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:CH<x>	Sets or queries the polarity setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:MATH<x>	Sets or queries the polarity setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:REF<x>	Sets or queries the polarity setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:CH<x>	Sets or queries the threshold high voltage level setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:MATH<x>	Sets or queries the threshold high voltage level setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:REF<x>	Sets or queries the threshold high voltage level setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:CH<x>	Sets or queries the threshold low voltage level setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:MATH<x>	Sets or queries the threshold low voltage level setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:REF<x>	Sets or queries the threshold low voltage level setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:WHEn	Sets or queries the condition setting for a transition trigger search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:TYPe	Sets or queries the trigger type setting for a search to determine where to place a mark ***NOTE.*** *Some trigger types are optional*
SEARCH:SEARCH<x>:TRIGger:A:WIDth:HIGHLimit	Sets or queries the upper time limit setting for a width search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WIDth:LOWLimit	Sets or queries the lower time limit setting for a width search to determine where to place a mark

Table 2-38: Search and Mark commands (cont.)

Command	Description
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:CH<x>	Sets or queries the polarity for a width search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:MATH<x>	Sets or queries the polarity for a width search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:REF<x>	Sets or queries the polarity for a width search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WIDth:WHEn	Sets or queries the width search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:EVENT	Sets or queries the window search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:CH<x>	Sets or queries the channel waveform threshold HIGH level for a window search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:MATH<x>	Sets or queries the math waveform threshold HIGH level for a window search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:REF<x>	Sets or queries the reference waveform threshold HIGH level for a window search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:CH<x>	Sets or queries the channel waveform threshold LOW level for a window search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:MATH<x>	Sets or queries the math waveform threshold LOW level for a window search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:REF<x>	Sets or queries the reference waveform threshold LOW level for a window search to determine where to place a mark
SEARCH:SEARCH<x>:TRIGger:A:WINdow:WIDth	Sets or queries the width setting for a window search to determine where to place a mark
SEARCH:STOP	Sets or queries the stop of acquisitions when a search finds an event

Status and Error command group

Use the commands in the Status and Error command Group to determine the status of the instrument and control events.

Several commands and queries used with the instrument are common to all devices on the GPIB bus. The IEEE Std 488.2-1987 defines these commands and queries. The common commands begin with an asterisk (*) character.

Table 2-39: Status and Error commands

Command	Description
*CLS	Clears status
*ESE	Sets or queries the bits in the Event Status Enable Register
*ESR?	Returns the contents of the Standard Event Status Register
*OPC	Generates the operation complete message in the standard event status register when all pending operations are finished Or returns “1” when all current operations are finished
*OPT?	Returns a list of options installed in the instrument
*PSC	Sets or queries the power on status flag
*PUD	Sets or queries a string of protected user data
*RST	Resets the instrument to factory default settings
*SRE	Sets or queries the bits in the Service Request Enable Register
*STB?	Returns the contents of the Status Byte Register
*WAI	Prevents the instrument from executing further commands until all pending operations finish
ALLEv?	Returns all events and their messages
BUSY?	Returns instrument status
DESE	Sets or queries the bits in the Device Event Status Enable Register
EVENT?	Returns event code from the event queue
EVMsg?	Returns event code, message from the event queue
EVQty?	Returns the number of events in the event queue
OPCEXtended	This command sets or queries the behavior of OPC commands and queries.

Trigger command group

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

There are two triggers: A and B. Where appropriate, the command set has parallel constructions for each trigger.

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 or B triggers to pulse or logic modes. With pulse triggering, the instrument triggers whenever it detects a pulse of a certain width or height. Logic triggering lets you logically combine the signals on one or more channels. The instrument then triggers when it detects a certain combination of signal levels. Video triggering enables you to trigger on the most common Standard Definition, High Definition, and custom video standards. You can trigger on all fields, all lines, or individual video lines.

NOTE. *Not available on some models.*

If you have Option MTH or MTM (Mask Testing) installed, you also have access to a communication mode. When this mode is selected, several communication trigger commands are available that work with the mask commands. Refer to Mask Command Group.

Table 2-40: Trigger commands (Not available on some models)

Command	Description
COUnter	Generates the reference, zeroes both the measured and derived statistical values, number of acquisitions used, and recalculates the A->B Sequence Trig on Nth Event value.
COUnter:DURation	Sets or queries the parameter that controls the schedule for updating the Log Table.
COUnter:GAIn	Sets or queries the Gain (scale) value for the Frequency or Time Interval applications.
COUnter:LOGAction	Clears all data from the Log Table.
COUnter:LOGNumber	Sets or queries the parameter that controls the schedule for updating the Log Table.
COUnter:LOGTable	Sets or queries the parameter that enables/disables the Log Table.
COUnter:REFerence	Sets or queries the Reference (offset) value for the Frequency or Time Interval applications.
COUnter:RESULTs?	Queries the measured and derived values obtained from the Frequency or Time Interval Applications.
COUnter:RESULTs:AVGmean?	Queries the derived values obtained from the Frequency or Time Interval Applications.

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
COUnter:RESULTs:DEViation?	Queries the derived values obtained from the Frequency or Time Interval Applications.
COUnter:RESULTs:MAXimum?	Queries the derived values obtained from the Frequency or Time Interval Applications.
COUnter:RESULTs:MINimum?	Queries the derived values obtained from the Frequency or Time Interval Applications.
COUnter:RESULTs:NUMber?	Queries the derived values obtained from the Frequency or Time Interval Applications.
COUnter:RESULTs:VALue?	Queries the measured value obtained from the Frequency or Time Interval Applications.
COUnter:SHOWLog	Sets or queries the parameter that controls the display of the Frequency or Time Interval Application Data.
COUnter:SHOWMeasurement	Sets or queries the parameter that controls the display of the Frequency or Time Interval Application Data.
COUnter:STAte	Enables the counter applications.
COUnter:TYPe	Sets or queries the Counter Application type (Frequency or Time Interval).
COUnter:UPDate	Sets or queries the parameter that controls the schedule for updating the Log Table.
COUnter:VIEW	Sets or queries the View of the Counter Trigger Source Frequency Application Data.
TRIGger	Forces a trigger event to occur or returns current trigger parameters for the instrument
TRIGger:A	Sets A trigger level to 50% or returns current A trigger parameters
TRIGger:{A|B}:EDGE:COUPling	Sets or queries the type of coupling for the edge trigger
TRIGger:{A|B}:EDGE:SLOpe	Sets or queries the slope for the edge trigger
TRIGger:{A|B}:EDGE:SLOpe:AUX	Sets or queries the slope for the edge trigger for the AUX In input
TRIGger:{A|B}:EDGE:SLOpe:CH<x>	Sets or queries the slope for the edge trigger for the channel
TRIGger:{A|B}:EDGE:SOUrce	Sets or queries the source for the edge trigger
TRIGger:{A|B}:LEVel	Sets or queries the level for the trigger
TRIGger:{A|B}:LEVel:CH<x>	Sets or queries the level for the trigger for the channel
TRIGger:{A|B}:EDGE?	Returns the source, coupling and source for the edge trigger
TRIGger:{A|B}:EDGE:COUPling:CH<x>	Sets or queries the type of coupling for the A or B trigger for the specified channel
TRIGger:{A|B}:LOGIc?	Returns all of the A or B trigger logic parameters
TRIGger:{A|B}:LOGIc:CLAss	Sets or queries the type of trigger logic
TRIGger:{A|B}:LOGIc:FUNCtion	Sets or queries the logical combination of the input channels for the logic trigger
TRIGger:{A|B}:LOGIc:INPut?	Returns the logic trigger input expected for channel 1, 2, and 3
TRIGger:{A|B}:LOGIc:INPut:ALL	Sets or queries the A or B logic trigger input condition for all the channels
TRIGger:{A|B}:LOGIc:INPut:CH<x>	Sets or queries the A or B logical trigger input for the specified channel
TRIGger:{A|B}:LOGIc:INPut:FORMat	Sets or queries the A or B logic trigger pattern format
TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>	Sets or queries the logic pattern trigger input for the indicated channel
TRIGger:{A|B}:LOGIc:PATtern:WHEn	Sets or queries the condition for generating the logic pattern trigger

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit	Sets or queries the maximum time that the selected pattern can be true and still generate a logic pattern trigger
TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit	Sets or queries the minimum time that the selected pattern can be true and still generate a logic pattern trigger
TRIGger:{A|B}:LOGIc:PATtern?	Returns the conditions for generating a logic pattern trigger
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE	Sets or queries the clock edge polarity for setup and hold triggering
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce	Sets or queries the clock source for the logic trigger setup and hold input
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold	Sets or queries the clock voltage threshold for setup and hold trigger
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold:CH<x>	Sets or queries the clock voltage threshold for setup and hold trigger for the channel
TRIGger:{A|B}:LOGIc:SETHold:CLOCk?	Returns clock edge polarity, voltage threshold, and source input for setup/hold triggering
TRIGger:{A|B}:LOGIc:SETHold:DATa:SOUrce	Sets or queries the data source for the setup and hold trigger
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold	Sets or queries the data voltage threshold for setup and hold trigger
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold:CH<x>	Sets or queries the data voltage threshold for setup and hold trigger for the channel
TRIGger:{A|B}:LOGIc:SETHold:DATa?	Returns the voltage threshold and data source for the setup/hold trigger
TRIGger:{A|B}:LOGIc:SETHold:HOLDTime	Sets or queries the hold time for the setup and hold violation triggering
TRIGger:{A|B}:LOGIc:SETHold:QUAlify	Sets or queries the setup and hold violation trigger qualifications
TRIGger:{A|B}:LOGIc:SETHold:SETTime	Sets or queries the setup time for setup and hold violation triggering
TRIGger:{A|B}:LOGIc:SETHold?	Returns clock edge polarity, voltage threshold and source; data voltage threshold and source; and setup/hold times for setup and hold violation triggering
TRIGger:{A|B}:LOGIc:STATE:CLOck:SOUrce	Sets or queries the A or B logic trigger state of clock source
TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x>	Sets or queries the slope for indicated channel of the logic state trigger
TRIGger:{A|B}:LOGIc:STATE:WHEn	Sets or queries the condition for generating a logic state trigger
TRIGger:{A|B}:LOGIc:STATE?	Returns the data input and trigger criteria for the logic state trigger
TRIGger:{A|B}:LOGIc:THReshold:CH<x>	Sets or queries the logic state trigger threshold voltage for the specified channel
TRIGger:{A|B}:LOWerthreshold:CH<x>	Sets or queries CH<x> lower trigger level for `TRIGger: LVLSrcpreference`
TRIGger:{A|B}:LOGIc:THReshold?	Returns the threshold voltage for all channels in logic trigger
TRIGger:{A|B}:PULse?	Returns the pulse trigger parameters
TRIGger:{A|B}:PULse:CLAss	Sets or queries the type of pulse on which to trigger
TRIGger:{A|B}:PULse:GLItch:LOWPASSfilter	Turns on or turns off the lowpass filter feature for Glitch trigger.
TRIGger:{A|B}:PULse:GLItch:POLarity	Sets or queries the polarity for the pulse glitch trigger
TRIGger:{A|B}:PULse:GLItch:POLarity:CH<x>	Sets or queries the polarity for the A or B pulse glitch trigger for the channel

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:{A|B}:PULse:GLItch:QUAlify	Sets or queries the glitch trigger qualifications
TRIGger:{A|B}:PULse:GLItch:TRIGIF	Sets or queries the acceptance or rejection of pulse glitch trigger, based on width
TRIGger:{A|B}:PULse:GLItch:WIDth	Sets or queries the width of the A pulse glitch trigger
TRIGger:{A|B}:PULse:GLItch?	Returns the current A glitch pulse trigger parameters
TRIGger:{A|B}:PULse:RUNT:POLarity	Sets or queries the polarity for the pulse runt trigger
TRIGger:{A|B}:PULse:RUNT:POLarity:CH<x>	Sets or queries the polarity for the A or B pulse runt trigger for the channel
TRIGger:{A|B}:PULse:RUNT:QUAlify	Sets or queries the polarity for the A or B pulse runt trigger
TRIGger:{A|B}:PULse:RUNT:THReshold:BOTh	Sets the upper and lower switching thresholds for the A pulse runt trigger
TRIGger:{A|B}:PULse:RUNT:THReshold:HIGH	Sets or queries the upper limit for the A pulse runt trigger
TRIGger:{A|B}:PULse:RUNT:THReshold:LOW	Sets or queries the lower limit for the A pulse runt trigger
TRIGger:{A|B}:PULse:RUNT:THReshold?	Returns the upper and lower thresholds for the A pulse runt trigger
TRIGger:{A|B}:PULse:RUNT:WHEn	Sets or queries the type of pulse width the trigger checks for when it uncovers a runt
TRIGger:{A|B}:PULse:RUNT:WIDth	Sets or queries the minimum width for A pulse runt trigger
TRIGger:{A|B}:PULse:RUNT?	Returns the current A runt pulse trigger parameters
TRIGger:{A|B}:PULse:SOUrce	Sets or queries the source for the A or B pulse trigger
TRIGger:{A|B}:PULse:TIMEOut:LOWPASSfilter	Turns on or turns off the lowpass filter feature for Timeout trigger.
TRIGger:{A|B}:PULse:TIMEOut:POLarity	Sets or queries the polarity for the A pulse timeout trigger
TRIGger:{A|B}:PULse:TIMEOut:POLarity:CH<x>	Sets or queries the polarity for the A or B pulse timeout trigger for the channel
TRIGger:{A|B}:PULse:TIMEOut:QUAlify	Sets or queries the timeout trigger qualifications
TRIGger:{A|B}:PULse:TIMEOut:TIMe	Sets or queries the pulse timeout trigger time
TRIGger:{A|B}:PULse:TIMEOut?	Returns the parameters for the A pulse timeout trigger
TRIGger:{A|B}:PULse:TRANsition:DELTATime	Sets or queries the delta time used in calculating the transition value
TRIGger:{A|B}:PULse:TRANsition:POLarity	Sets or queries the polarity for the A pulse transition trigger
TRIGger:{A|B}:PULse:TRANsition:POLarity:CH<x>	Sets or queries the polarity for the A or B pulse transition trigger for the channel
TRIGger:{A|B}:PULse:TRANsition:QUAlify	Sets or queries the transition trigger qualifications
TRIGger:{A|B}:PULse:TRANsition:THReshold:BOTh	Sets the upper and lower thresholds for the transition trigger
TRIGger:{A|B}:PULse:TRANsition:THReshold:HIGH	Sets or queries the upper transition trigger threshold
TRIGger:{A|B}:PULse:TRANsition:THReshold:LOW	Sets or queries the lower transition trigger threshold

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:{A|B}:PULse:TRANsition:THReshold?	Returns the upper and lower threshold limits for the transition time trigger
TRIGger:{A|B}:PULse:TRANsition:WHEn	Sets or queries whether to check for a transitioning signal that is faster or slower than the specified delta time for the transition time trigger
TRIGger:{A|B}:PULse:TRANsition?	Returns the delta time, polarity, and both upper and lower threshold limits for the transition time trigger
7TRIGger:{A|B}:PULse:WIDth:HIGHLimit	Sets or queries the upper limit for the A pulse width trigger
TRIGger:{A|B}:PULse:WIDth:LOWLimit	Sets or queries the lower limit for the pulse width trigger
TRIGger:{A|B}:PULse:WIDth:LOWPASSfilter	Turns on or turn off the lowpass filter feature for pulse width trigger.
TRIGger:{A|B}:PULse:WIDth:POLarity	Sets or queries the polarity for the pulse width trigger
TRIGger:{A|B}:PULse:WIDth:POLarity:CH<x>	Sets or queries the polarity for the A or B pulse width trigger for the channel
TRIGger:{A|B}:PULse:WIDth:QUAlify	Sets or queries the pulse width trigger qualifications
TRIGger:{A|B}:PULse:WIDth:WHEn	Sets or queries the criteria for width specification of pulse width trigger events
TRIGger:{A|B}:PULse:WIDth?	Returns the trigger pulse width parameters
TRIGger:{A|B}:PULse:WINdow?	Returns the current window pulse trigger parameters
TRIGger:{A|B}:PULse:WINdow:EVENT	Sets or queries the window trigger event
TRIGger:{A|B}:PULse:WINdow:POLarity	Sets or queries the pulse trigger window polarity of the selected trigger Source
TRIGger:{A|B}:PULse:WINdow:POLarity:CH<x>	Sets or queries the pulse trigger window polarity of the specified channel
TRIGger:{A|B}:PULse:WINdow:QUAlify	Sets or queries the window trigger qualifications
TRIGger:{A|B}:PULse:WINdow:THReshold:BOTh	Sets both window thresholds
TRIGger:{A|B}:PULse:WINdow:THReshold:HIGH	Sets or queries high window threshold
TRIGger:{A|B}:PULse:WINdow:THReshold:LOW	Sets or queries low window threshold
TRIGger:{A|B}:PULse:WINdow:THReshold?	Returns the upper and lower thresholds for the pulse window trigger
TRIGger:{A|B}:PULse:WINdow:WIDTH	Sets or queries minimum width of violation
TRIGger:{A|B}:READY?	Queries the trigger ready state
TRIGger:{A|B}:TYPe	Sets or queries the type of A or B trigger
TRIGger:{A|B}:UPPerthreshold:CH<x>	Sets or queries the CH<x> upper trigger level for TRIGger:LVLSrcpreference
TRIGger:A:BUS:CAN:CONDition	Sets or queries the CAN bus trigger condition
TRIGger:A:BUS:CAN:DATa:DIRection	Sets or queries the CAN bus trigger data direction
TRIGger:A:BUS:CAN:DATa:FORMat	Sets or queries the CAN bus trigger data format
TRIGger:A:BUS:CAN:DATa:QUALifier	Sets or queries CAN bus trigger data qualifier
TRIGger:A:BUS:CAN:DATa:SIZe	Sets or queries the CAN bus trigger data size
TRIGger:A:BUS:CAN:DATa:VALue	Sets or queries CAN bus trigger data value
TRIGger:A:BUS:CAN:FRAMEtype	Sets or queries CAN bus trigger frame type

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:DIRection	Sets or queries sets the CAN bus trigger identifier direction
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:FORMat	Sets or queries the CAN bus trigger identifier format
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:MODe	Sets or queries CAN bus trigger identifier mode
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:VALue	Sets or queries the CAN bus trigger identifier value
TRIGger:A:BUS:DATa:FORMat	Sets or queries the trigger data format for the bus
TRIGger:A:BUS:DATa:VALue	Sets or queries the trigger data value depending on the format selected for the bus
TRIGger:A:BUS:ETHERnet:CONDition	Specifies a field or condition within an Ethernet frame to trigger on
TRIGger:A:BUS:ETHERnet:DATa:FORMat	This command and query sets the format of the data to either binary or hex.
TRIGger:A:BUS:ETHERnet:DATa:OFFSet	Specifies the data offset value, in bytes, to use when triggering on the Ethernet data field
TRIGger:A:BUS:ETHERnet:DATa:SIZe	Specifies the number of contiguous TCP/IPv4/MAC client bytes to use when triggering on the Ethernet data field
TRIGger:A:BUS:ETHERnet:DATa:VALue	Specifies the binary value to use when triggering on the Ethernet binary data field
TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue	Specifies the 32-bit IPv4 value to use when triggering on the Ethernet IPv4 header address destination field
TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet IPv4 header address destination field
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue	Specifies the 8-bit binary value to use when triggering on the Ethernet IPv4 header protocol field
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat	Specifies the data format (binary or hex) of the bit pattern to use when triggering on the Ethernet IPv4 header protocol field
TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue	Specifies the 32-bit value to use when triggering on the Ethernet IPv4 header address source field
TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat	Specifies the data format (binary or hex) of the bit pattern to use when triggering on the Ethernet IPv4 header address source field
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue	Specifies the 48-bit value to use when triggering on the Ethernet MAC address destination field
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat	Specifies the data format (binary or hex) of the bit pattern to use when triggering on the Ethernet MAC address destination field
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet MAC address source field
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue	Specifies the 48-bit value to use when triggering on the Ethernet MAC address source field
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue	Specifies the 16-bit value to use when triggering on the Ethernet MAC length/type
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet MAC length or type

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:A:BUS:ETHERnet:QTAG:VALue	Specifies the 32-bit value to use when triggering on the Ethernet Q-Tag field
TRIGger:A:BUS:ETHERnet:QTAG:FORMat	Specifies the data format (binary or hex) to use when triggering on the Q-Tag field
TRIGger:A:BUS:ETHERnet:QUALifier	Specifies the qualifier to be used with the Ethernet trigger condition is set to MACLENgth or DATa
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue	Specifies the 32-bit value to use when triggering on the Ethernet TCP header acknowledgement number
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header acknowledgement number
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue	Specifies the 16-bit value to use when triggering on the Ethernet TCP header destination port number
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header destination port number
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue	Specifies the 32-bit value to use when triggering on the Ethernet TCP header sequence number
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header sequence number
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue	Specifies the 16-bit value to use when triggering on the Ethernet TCP header source port number
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header source port number
TRIGger:A:BUS:FLEXRAY:CONDition	Sets or queries the FLEXRAY bus trigger condition
TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat	Sets or queries FLEXRAY bus trigger cycle count format
TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier	Sets or queries FLEXRAY bus trigger cycle count qualifier
TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue	Sets or queries the FLEXRAY bus trigger cycle count value
TRIGger:A:BUS:FLEXRAY:DATa:FORMat	Sets or queries the FLEXRAY bus trigger data format
TRIGger:A:BUS:FLEXRAY:DATa:OFFSet	Sets or queries the FLEXRAY bus trigger data offset
TRIGger:A:BUS:FLEXRAY:DATa:QUALifier	Sets or queries the FLEXRAY bus trigger data qualifier
TRIGger:A:BUS:FLEXRAY:DATa:SIZe	Sets or queries the FLEXRAY bus trigger data size
TRIGger:A:BUS:FLEXRAY:DATa:VALue	Sets or queries the FLEXRAY bus trigger data value
TRIGger:A:BUS:FLEXRAY:EOFTYPE	Sets or queries the FLEXRAY bus trigger end of file type
TRIGger:A:BUS:FLEXRAY:ERRTYPE	Sets or queries the FLEXRAY bus trigger error type
TRIGger:A:BUS:FLEXRAY:FRAMEType	Sets or queries the FLEXRAY bus trigger frame type
TRIGger:A:BUS:FLEXRAY:HEADER:CRC	Sets or queries the FLEXRAY bus trigger header CRC
TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount	Sets or queries the FLEXRAY bus trigger header cycle count
TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID	Sets or queries the FLEXRAY bus trigger header frame ID

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:A:BUS:FLEXRAY:HEADER:INDBits	Sets or queries the FLEXRAY bus trigger header Indicator Bits
TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth	Sets or queries the FLEXRAY bus trigger header Payload Length
TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat	Sets or queries the FLEXRAY bus trigger identifier format
TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier	Sets or queries the FLEXRAY bus trigger identifier qualifier
TRIGger:A:BUS:FLEXRAY:IDentifier:VALue	Sets or queries the FLEXRAY bus trigger identifier value
TRIGger:A:BUS:I2C:ADDress:FORMat	Sets or queries the address format for the I2C bus
TRIGger:A:BUS:I2C:ADDress:MODe	Sets or queries the address mode for the I2C bus
TRIGger:A:BUS:I2C:ADDress:VALue	Sets or queries the binary address string used for the I2C trigger of the bus
TRIGger:A:BUS:I2C:CONDition	Sets or queries the trigger condition for the I2C bus
TRIGger:A:BUS:I2C:DATa:DIRection	Sets or queries the data direction for the I2C bus
TRIGger:A:BUS:LIN:CONDition	Sets or queries the LIN bus trigger condition
TRIGger:A:BUS:LIN:DATa:FORMat	Sets or queries the LIN bus trigger data format
TRIGger:A:BUS:LIN:DATa:QUALifier	Sets or queries the LIN bus trigger data qualifier
TRIGger:A:BUS:LIN:DATa:SIZe	Sets or queries the LIN bus trigger data size
TRIGger:A:BUS:LIN:DATa:VALue	Sets or queries the LIN bus trigger data value
TRIGger:A:BUS:LIN:ERRTYPE	Sets or queries the LIN bus trigger error type
TRIGger:A:BUS:LIN:IDentifier:FORMat	Sets or queries the LIN bus trigger identifier format
TRIGger:A:BUS:LIN:IDentifier:VALue	Sets or queries the LIN trigger identifier value
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat	Sets the data format of the address (either binary or hex) to use in a MIL-STD-1553 bus command word trigger
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier	Specifies the qualifier to use when triggering on a MIL-STD-1553 bus command word remote terminal address field
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue	Specifies the value of the 5-bit remote terminal address to use in a MIL-STD-1553 bus command word trigger
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt	Specifies the bit pattern for the 5-bit Word Count/Mode Code sub-address field to use in a MIL-STD-1553 bus command word trigger
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat	Specifies the data format (binary or hex) of the bit pattern for the 5-bit Word Count/Mode Code field to use in a MIL-STD-1553 bus command word trigger
TRIGger:A:BUS:MIL1553B:COMMAND:PARity	Specifies the parity to use in a MIL-STD-1553 bus command word trigger
TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress	Specifies the 5 bit sub-address to use in a MIL-STD-1553 bus command word sub-address trigger
TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:FORMat	Specifies the data format (binary or hex) of the sub-address field bit pattern to use in a MIL-STD-1553 bus command word sub-address trigger
TRIGger:A:BUS:MIL1553B:COMMAND:TRBit	Specifies a trigger on either a “transmit” or a “receive” signal for the transmit/receive bit (bit 9) for a MIL-STD-1553 bus command word trigger
TRIGger:A:BUS:MIL1553B:CONDition	Specifies which word type or condition within a MIL-STD-1553 bus word to trigger on

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:A:BUS:MIL1553B:DATA:FORMat	Specifies the format (binary or hex) of the bit pattern to use for a MIL-STD-1553 bus data word trigger
TRIGger:A:BUS:MIL1553B:DATA:PARity	Specifies the parity to use when triggering on a MIL-STD-1553 bus data word parity bit
TRIGger:A:BUS:MIL1553B:DATA:VALue	Specifies the value to use for a MIL-STD-1553 bus data word trigger
TRIGger:A:BUS:MIL1553B:ERRTYPE	Specifies the signaling error type to use for a MIL-STD-1553 bus data word trigger: Parity, Sync, Manchester, or Data
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:FORMat	Specifies the format (binary or hex) of the bit pattern to use when triggering on the MIL-STD-1553 status address field
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier	Specifies the qualifier to use when triggering on the MIL-STD-1553 status address field
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue	Specifies the value of the 5-bit remote terminal address to use when triggering on the MIL-STD-1553 status address field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR	Specifies the status word broadcast command received (BCR) bit value (bit 15) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY	Specifies the status word busy bit value (bit 16) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA	Specifies the status word dynamic bus control acceptance (DBCA) bit value (bit 18) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR	Specifies the status word instrumentation bit value (bit 10) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME	Specifies the status word message error bit value (bit 9) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity	Specifies the parity to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ	Specifies the status word service request (SRQ) bit value (bit 11) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF	Specifies the status word subsystem flag bit value (bit 17) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF	Specifies the status word terminal flag bit value (bit 19) to use when triggering on the MIL-STD-1553 status bit field
TRIGger:A:BUS:MIL1553B:TIME:LESSLimit	Specifies either the minimum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit or the minimum inter-message gap (IMG)
TRIGger:A:BUS:MIL1553B:TIME:MORELimit	Specifies either the maximum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit or the maximum inter-message gap (IMG)
TRIGger:A:BUS:MIL1553B:TIME:QUALifier	Specifies the qualifier to use when triggering on either the remote terminal response time (RT) or the inter-message gap (IMG)
TRIGger:A:BUS:PCIE:CHARacter:CHAR	Sets or queries the PCIE bus trigger character
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus	Sets or queries the PCIE bus trigger character symbol minus

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS	Sets or queries PCIE bus trigger character symbol plus
TRIGger:A:BUS:PCIE:CONDition	Sets or queries the PCIE bus trigger condition
TRIGger:A:BUS:PCIE:DISParity	Sets or queries the PCIE bus trigger disparity
TRIGger:A:BUS:PCIE:ERROR	Sets or queries the PCIE bus trigger error
TRIGger:A:BUS:PCIE:FORMat	Sets or queries the PCIe bus trigger format
TRIGger:A:BUS:PCIE:PATtern:CHAR<x>	Sets or queries the PCIE bus trigger pattern character
TRIGger:A:BUS:PCIE:PATtern:ORDERedset	Sets or queries the PCIE bus trigger pattern ordered set
TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>	Sets or queries the PCIE bus trigger pattern symbol minus
TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>	sets or queries the PCIE bus trigger pattern plus symbol
TRIGger:A:BUS:RS232C:CONDition	Sets or queries the trigger condition for the specified RS232C bus
TRIGger:A:BUS:S8B10B:CHARacter:CHAR	Sets or queries the 8B10B character when the Trigger Condition is set to Character
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus	Sets or queries the 8B10B character (negative disparity) when the Trigger Condition is set to Character
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS	Sets or queries the Character value (positive disparity view) when the Trigger Condition is set to Character
TRIGger:A:BUS:S8B10B:CONDition	Sets or queries the trigger condition for the 8B10B bus
TRIGger:A:BUS:S8B10B:DISParity	Sets or queries the type of disparity that the 8B10B bus will trigger on
TRIGger:A:BUS:S8B10B:ERROR	Sets or queries whether the oscilloscope will trigger on a character error or a disparity error
TRIGger:A:BUS:S8B10B:FORMat	Sets or queries the pattern editing format
TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>	Sets or queries the specified 8B10B character when the Trigger Condition is set to Pattern
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>	Sets or queries the specified 8B10B character (negative disparity) when the Trigger Condition is set to Pattern
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>	Sets or queries the specified 8B10B character (positive disparity) when the Trigger Condition is set to Pattern
TRIGger:A:BUS:SOUrce	Sets or queries the source for a bus trigger
TRIGger:A:BUS:SPI:CONDition	Sets or queries the trigger condition for the SPI bus
TRIGger:A:BUS:USB:CONDition	Sets or queries the trigger condition for the specified USB bus
TRIGger:A:BUS:USB:DATa:HIVALue	Sets or queries the trigger data value for data token of the specified USB bus
TRIGger:A:BUS:USB:DATa:OFFSet	Sets or queries the data offset for the USB bus trigger
TRIGger:A:BUS:USB:DATa:SIZe	Sets or queries the number of contiguous data bytes to USB bus trigger
TRIGger:A:BUS:USB:DATa:TYPe	Sets or queries the data packet type when USB bus trigger condition is Data Packet
TRIGger:A:BUS:USB:DATa:VALue	Sets or queries the USB bus trigger data value for data token
TRIGger:A:BUS:USB:ENDPoint:HIVALue	Sets or queries the endpoint value for data token of the specified USB bus
TRIGger:A:BUS:USB:ENDPoint:VALue	Sets or queries the endpoint value for normal token of the specified USB bus

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:A:BUS:USB:ERRTYPE	Set or queries the error type when USB bus trigger on condition is set to Error
TRIGger:A:BUS:USB:HANDShaketype	Set or queries the handshake type when USB bus trigger on condition is set to Handshake Packet
TRIGger:A:BUS:USB:QUAlifier	Set or queries the USB bus trigger qualifier for address, endpoint, and data
TRIGger:A:BUS:USB:SPECIALType	Set or queries the PID value for Special Packet USB bus trigger on condition
TRIGger:A:BUS:USB:SPLIT:ET:VALue	Set or queries the End Point Type value of the specified USB bus trigger
TRIGger:A:BUS:USB:SPLIT:HUB:VALue	Set or queries the port address for SPLIT token of the USB bus trigger
TRIGger:A:BUS:USB:SPLIT:PORT:VALue	Set or queries the port address for SPLIT token of the USB bus trigger
TRIGger:A:BUS:USB:SPLIT:SC:VALue	Set or queries the Start/Complete value for the USB bus trigger on split token field
TRIGger:A:BUS:USB:SPLIT:SE:VALue	Set or queries the Start/End value for the USB bus trigger for the specified USB bus
TRIGger:A:BUS:USB:TOKENType	Set or queries the Token Type when Trigger condition is set to TOKEN for the specified USB bus
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:PULSEForm	Sets or queries the selected communication pulse form for the specified USB bus
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:HIGH	Sets the communication threshold high level
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:LOW	Sets the communication threshold low level
TRIGger:A:COMMunication:BITRate	Sets or queries the communication bit rate
TRIGger:A:COMMunication:CLOCk:POLarity	Sets or queries the polarity of the clock signal
TRIGger:A:COMMunication:CMI:PULSEForm	Selects the CMI pulse form
TRIGger:A:COMMunication:CODe	Sets the communication signal code
TRIGger:A:COMMunication:SOUrce	Sets the communication source channel
TRIGger:A:COMMunication:SOUrce:TYPe	Sets the type of communication source channel
TRIGger:A:COMMunication:STANdard	Selects the communication standard that identifies the code and bit rate
TRIGger:A:EDGE:ENVelope	Sets or queries whether the A edge trigger will use envelope triggering.
TRIGger:A:HOLDoff:ACTUal?	Returns the holdoff time actually used by the A trigger
TRIGger:A:HOLDoff:BY	Sets or queries the type of holdoff for the A trigger
TRIGger:A:HOLDoff:TIMe	Sets or queries the A trigger holdoff time
TRIGger:A:HOLDoff?	Returns the A trigger holdoff parameters
TRIGger:A:MODe	Sets or queries the A trigger mode
TRIGger:A:PULse:GLItch:ENVelope	Sets or queries whether the A glitch trigger will use envelope triggering.
TRIGger:A:PULse:PERiod?	Queries the Pulse Trigger settings specific to Period/Frequency trigger.
TRIGger:A:PULse:PERiod:HIGHLimit	Sets or queries the Period/Frequency high limit.
TRIGger:A:PULse:PERiod:LOWLimit	Sets or queries the Period/Frequency low limit.
TRIGger:A:PULse:PERiod:POLarity	Sets or queries the polarity for the Pulse Period trigger.

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:A:PULse:PERiod:QUAlify	Sets or queries the Qualification setup for Pulse Period Trigger.
TRIGger:A:PULse:PERiod:VIEW	Sets or queries the “view” for the pulse period trigger.
TRIGger:A:PULse:PERiod:WHEn	Sets or queries the “trigger when” enumeration for the Pulse Period Trigger.
TRIGger:A:PULse:TIMEOut:ENVelope	Sets or queries whether the A timeout trigger will use envelope triggering.
TRIGger:A:PULse:WIDth:ENVelope	Sets or queries whether the B width trigger will use envelope triggering.
TRIGger:A:SERIAL:BITRate	Sets or queries the clock/data bit rate
TRIGger:A:SERIAL:CLOCk:LEVel	Sets or queries the serial trigger clock level
TRIGger:A:SERIAL:CLOCk:POLarity	Sets or queries the serial clock polarity
TRIGger:A:SERIAL:CLOCk:SOUrce	Sets or queries the serial data source channel
TRIGger:A:SERIAL:CODe	Sets or queries the signal code
TRIGger:A:SERIAL:DATa:FORMat	Sets or queries how the Pattern string is formatted
TRIGger:A:SERIAL:DATa:PATtern	Sets or queries the data pattern to allow up to 64 bit serial patterns
TRIGger:A:SERIAL:DATa:PATtern:NRZ	Sets or queries the data pattern to allow up to 64 bit serial patterns
TRIGger:A:SERIAL:DATa:PATtern:S8B10B	Sets or queries the data pattern to allow up to 64 bit serial patterns
TRIGger:A:SERIAL:LOCKLen	Sets or queries the length in bits of the repeating bit pattern for pattern lock trigger
TRIGger:A:SERIAL:LOCKOffset	Sets or queries the length in bits of the repeating bit pattern for pattern lock trigger
TRIGger:A:SERIAL:SOUrce	Sets or queries the length in bits of the repeating bit pattern for pattern lock trigger
TRIGger:A:SERIAL:STANdard	Sets or queries the length in bits of the repeating bit pattern for pattern lock trigger
TRIGger:A:SERIAL:TRIGgeron	Sets or queries the length in bits of the repeating bit pattern for pattern lock trigger
TRIGger:A:VIDeo?	Returns the video parameters for the A trigger
TRIGger:A:VIDeo:CUSTom?	Returns the custom video parameters for the A trigger
TRIGger:A:VIDeo:CUSTom:FORMat	Sets or queries the video scan format when the video standard is set to Custom
TRIGger:A:VIDeo:CUSTom:LINEPeriod	Sets or queries the custom video line period
TRIGger:A:VIDeo:CUSTom:SYNCInterval	Sets or queries the custom video sync interval
TRIGger:A:VIDeo:FIELD	Sets or queries the video field trigger
TRIGger:A:VIDeo:HOLdoff:FIELD	Sets or queries the trigger holdoff in video field units
TRIGger:A:VIDeo:LINE	Sets or queries the trigger delay as several video lines
TRIGger:A:VIDeo:POLarity	Sets or queries the polarity of the video trigger
TRIGger:A:VIDeo:SCAN	Sets or queries the video scan rate
TRIGger:A:VIDeo:SOUrce	Sets or queries the video trigger source
TRIGger:A:VIDeo:STANdard	Sets or queries the video standard
TRIGger:AUXLevel	Sets or queries the level for the auxiliary trigger
TRIGger:B	Sets the B trigger level to 50% or queries the B trigger parameters
TRIGger:B:BY	Sets or queries B trigger time or event qualifiers
TRIGger:B:EDGE:ENVelope	Sets or queries whether the B edge trigger will use envelope triggering.
TRIGger:B:EVENTS:COUNt	Sets or queries the number of events that must occur before the B trigger occurs
TRIGger:B:EVENTS?	Returns the current B trigger events parameter

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:B:PULse:GLItch:ENVelope	Sets or queries whether the B glitch trigger will use envelope triggering.
TRIGger:B:PULse:PERiod?	Queries the Pulse Trigger settings specific to Period/Frequency trigger.
TRIGger:B:PULse:PERiod:LOWLimit	Sets or queries the Period/Frequency low limit.
TRIGger:B:PULse:PERiod:POLarity	Sets or queries the polarity for the Pulse Period trigger.
TRIGger:B:PULse:PERiod:QUAlify	Sets or queries the Qualification setup for Pulse Period Trigger.
TRIGger:B:PULse:PERiod:VIEW	Sets or queries the “view” for the pulse period trigger.
TRIGger:B:PULse:PERiod:WHEn	Sets or queries the “trigger when” enumeration for the Pulse Period Trigger.
TRIGger:B:PULse:TIMEOut:ENVelope	Sets or queries whether the B timeout trigger will use envelope triggering.
TRIGger:B:PULse:WIDth:ENVelope	Sets or queries whether the B width trigger will use envelope triggering.
TRIGger:B:RESET:ACCEPTCOUNT?	Queries the Reset Accept Count for a sequence trigger reset by the ACCept and REJect Reset Types.
TRIGger:B:RESET:ACCEPTTIMEout	Sets or queries the Reset Accept Time Out timer for a sequence trigger reset by the ACCept Reset Type.
TRIGger:B:RESET:RATIO?	Queries the (floating point) Ratio of accepted to total triggers for a sequence trigger reset by the ACCept and REJect Reset Types.
TRIGger:B:RESET:RATIOENable	Sets or queries the Ratio Enable enumerated value (OFF/ON) to display in the graticule are the ratio of accepted to total triggers for a sequence trigger reset by the ACCept and REJect Reset Types.
TRIGger:B:RESET:REJECTCOUNT?	Queries the Reset Reject Count for a sequence trigger reset by the ACCept and REJect Reset Types.
TRIGger:B:RESET:REJECTTIMEout	Sets or queries the Reset Reject Time Out timer for a sequence trigger reset by the REJect Reset Type.
TRIGger:B:RESET:SOUrce	Sets or queries the source for the trigger reset
TRIGger:B:RESET:STATE	Sets or queries the state of the B trigger
TRIGger:B:RESET:THReshold	Sets or queries the threshold for the trigger reset
TRIGger:B:RESET:TIMEOut	Sets or queries the reset time for the timeout trigger reset
TRIGger:B:RESET:TOTALCOUNT?	Queries the Reset Total Count for a sequence trigger reset by the ACCept and REJect Reset Types.
TRIGger:B:RESET:TRANsition	Sets or queries the type of transition required for a transition trigger reset
TRIGger:B:RESET:TYPe	Sets or queries the type of A→B sequential trigger reset
TRIGger:B:SCAN:ADVANCEafter	Sets or queries the B-Event Scan advance after value
TRIGger:B:SCAN:ENAble	Sets or queries if the B-Event Scan is on or off
TRIGger:B:SCAN:ENDevent	Sets or queries the end event value of B-Event Scan
TRIGger:B:SCAN:MODE	Sets or queries the sequence advance mode of B-Event Scan
TRIGger:B:SCAN:STARTevent	Sets or queries the start event value of B-Event Scan
TRIGger:B:STATE	Returns the current state of the triggering system
TRIGger:B:TIMe	Sets or queries the B trigger delay time
TRIGger:ENHanced	Sets or queries the state of trigger position enhancement
TRIGger:LVLSrcpreference	Sets or queries how the trigger source and levels interact

Table 2-40: Trigger commands (Not available on some models) (cont.)

Command	Description
TRIGger:QUALification:BUS:FORMat	Sets or queries the format of the bit pattern for the bus qualification
TRIGger:QUALification:BUS:SOUrce	Sets or queries the source for the bus qualification
TRIGger:QUALification:BUS:VALue	Sets or queries the bit pattern for the bus qualification
TRIGger:SENSITivity	Set or queries the state of trigger sensitivity. When on, the instrument calculates a rolling average for trigger position enhancement
TRIGger:STATE?	Returns the current state of the triggering system
Visual trigger commands (available with Option VET)	
TRIGger:EQUation<1..1>	Sets the Visual Trigger Equation string, which defines the behavior of the areas
TRIGger:SHOWEQuation	Causes the Visual Trigger Equation to be displayed in the upper left portion of the oscilloscope graticule
VISual?	Returns the settings for each visual trigger area
VISual:AREA<x>?	Returns the settings for the specified visual trigger area
VISual:AREACOLOr	Sets or queries the colors used by visual trigger areas
VISual:AREA<x>:DISplay	Causes the areas to be hidden or displayed on the oscilloscope screen Queries the status of the area display
VISual:AREA<x>:FLIP	Flips the specified visual trigger area
VISual:AREA<x>:HEIGHT	Sets or queries the height of the specified visual trigger area
VISual:AREA<x>:LABel	Sets or queries the label of the specified visual trigger area
VISual:AREA<x>:OPERATION<1..1>	Sets or queries the operation of the area specified by <x>
VISual:AREA<x>:ROTAtion	Sets or queries the rotation angle of the specified visual trigger area
VISual:AREA<x>:SHAPE	Selects or queries the initial shape of area <x>
VISual:AREA<x>:SHOWLOGic	Causes area Source and Operation indicators to be hidden or displayed on screen. Queries the status of this feature
VISual:AREA<x>:SOURCE<1..1>	Selects or queries the signal source for the area <x>. The source can only be an analog channel
VISual:AREA<x>:VERTICES	Sets or queries the polygon vertex x and y coordinate values for an area
VISual:AREA<x>:WIDTH	Sets or queries the width of the specified visual trigger area
VISual:AREA<x>:XPOSition	Sets or queries horizontal position of specified visual trigger area
VISual:AREA<x>:YPOSition	Sets or queries vertical position of the specified visual trigger area
VISual:ASPECTratio	Sets or queries aspect ratio setting of the visual trigger system
VISual:DELETEAREA	Deletes the specified visual trigger area
VISual:ENAble	Enables or disables the visual triggering. Queries the status (on or off) of the visual triggering
VISual:FILE:RECALL	Recalls a file of Visual Trigger Parameters, updating the database appropriately
VISual:FILE:SAVE	Saves only the Visual Trigger Parameters, to a file that you name in the <string>

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 control the display of channel, reference, and math waveforms.

You can replace VOLts with SCAle in the vertical commands. This provides program compatibility with earlier models of Tektronix instruments.

Table 2-41: Vertical commands

Command	Description
AUXIn:BANdwidth	Sets or queries the selectable low-pass bandwidth limit filter of the AUX In input
AUXIn:COUPling	Sets or queries the input attenuator coupling setting for the AUX In input
AUXIn:OFFSet	Sets or queries the vertical offset for the AUX In input
AUXIn:PRObe:AUTOZero	Sets the probe attached to the AUX In input to autozero
AUXIn:PRObe:DEGAUSS	Starts a degauss cycle of the probe attached to the AUX In input
AUXIn:PRObe:DEGAUSS:STATE?	Returns whether the probe attached to the AUX In input is degaussed
AUXIn:PRObe:FORCEDRange	Sets the probe attached to the AUX In input to the default range of the probe, or the command queries the default range of the probe
AUXIn:PRObe:GAIN?	Returns the gain factor of the probe that is attached to the AUX In input
AUXIn:PRObe:ID:SERnumber?	Returns the probe serial number
AUXIn:PRObe:ID:TYPe?	Returns the probe type
AUXIn:PRObe:INPUTMode	Sets or queries one of the probe's four input modes
AUXIn:PRObe:INPUTMode:AOFFSet	Sets or queries the requested A mode offset control of the probe that is attached to the AUX In input
AUXIn:PRObe:INPUTMode:BOFFSet	Sets or queries the requested B mode offset control of the probe that is attached to the AUX In input
AUXIn:PRObe:INPUTMode:CMOFFSet	Sets or queries the requested common mode offset control of the probe that is attached to the AUX In input
AUXIn:PRObe:INPUTMode:DMOFFSet	Sets or queries the requested differential mode offset control of the probe that is attached to the AUX In input
AUXIn:PRObe:RANge	Sets or queries the set attenuation range of the probe on the AUX In input
AUXIn:PRObe:RESistance?	Returns the resistance factor of the probe that is attached to the AUX In input
AUXIn:PRObe:SET	Sets or queries aspects of probe accessory user interfaces, for example probe attenuation factors
AUXIn:PRObe:SIGnal	Sets or queries aspects of probe accessory user interfaces
AUXIn:PRObe:TIPtype	Sets or queries the type of probe tip being used
AUXIn:PRObe:UNIts?	Returns a string describing the units of measure for the probe attached to the AUX In input
AUXIn:PROBEFunc:EXTAtten	Sets or queries the input-output ratio, of external attenuation or gain, between the signal and the AUX In input
AUXIn:PROBEFunc:EXTDBatten	Sets or queries the input-output ratio (expressed in decibel units) of external attenuation or gain between the signal and the instrument AUX In input

Table 2-41: Vertical commands (cont.)

Command	Description
AUXIn:PROBEFunc:EXTUnits	Sets or queries the unit of measurement for the external attenuator of the AUX In input
AUXIn:VTERm:DUAL:A	Sets or queries the termination voltage for probes with dual inputs that support settable termination voltage
AUXIn:VTERm:DUAL:B	Sets or queries the termination voltage for probes with dual inputs that support settable termination voltage
CH<x>?	Returns vertical parameters for the specified channel
CH<x>:ATIACTive?	Returns whether the specified channel is an ATI channel.
CH<x>:AVAILable?	Returns whether the specified channel is available.
CH<x>:BANdwidth	Sets or queries the bandwidth of the specified channel
CH<x>:BANdwidth:ENHanced	Sets or queries either the “Bandwidth Enhancement (DSP) Enabled” or “Analog Only” option
CH<x>:BANdwidth:ENHanced:APPLYtoall	Sets or queries whether the enhanced bandwidth filter applies to all active channels
CH<x>:BANdwidth:ENHanced:FORCe	Sets or queries the Force constant sample rate of the Digital Filters (DSP)
CH<x>:BANdwidth:ENHanced:STATE?	Returns the state of the bandwidth enhancement filter
CH<x>:COUPling	Sets or queries the coupling setting for the specified channel
CH<x>:DESKew	Sets or queries the deskew time for the specified channel
CH<x>:FASTAcqcapable?	Returns whether the specified channel is capable of FastAcq.
CH<x>:FASTFRamecapable?	Returns whether the specified channel is capable of FastFrame acquisitions.
CH<x>:ICAPture:SOUrce	Selects which digital probe signal will acquire for the specified channel when put in iCapture mode ***NOTE.*** *Not available on some models*
CH<x>:ICAPture:STATE	Selects the state of the digital probe signal will acquire for the specified channel when put in iCapture mode ***NOTE.*** *Not available on some models*
CH<x>:INVert	Sets or queries the invert function for the specified channel.
CH<x>:LABel:NAMe	Defines or queries the label for the channel waveform
CH<x>:LABel:XPOS	Sets or queries the X display coordinate for the channel waveform label
CH<x>:LABel:YPOS	Sets or queries the Y display coordinate for the channel waveform label
CH<x>:OFFSet	Sets or queries the channel offset
CH<x>:OPTIcal:RCVR	Sets or queries the Optical Reference Receiver DSP filter
CH<x>:OPTIcal:WLENgth	Sets or queries the optical wavelength (in nanometers) that is used by the Probe and Scope to compensate for the applied Optical signal.
CH<x>:OPTIcal:WLENgth:LIST?	Returns the comma-separated list of wavelengths with calibrated responses from the Probe.
CH<x>:POSition	Sets or queries the channel vertical position
CH<x>:PRObe?	Returns the gain, resistance, units, and ID of the probe that is attached to the specified channel

Table 2-41: Vertical commands (cont.)

Command	Description
CH<x>:PRObe:AUTOZero	Autozeros the probe attached to the specified channel
CH<x>:PRObe:DEGAUSS	Starts a degauss cycle of the probe attached to the specified channel
CH<x>:PRObe:DEGAUSS:STATE?	Returns whether the probe attached to the specified channel is degaussed
CH<x>:PRObe:FORCEDRange	Sets the attached probe to its default range, or it queries the default range of the probe
CH<x>:PRObe:GAIN?	Returns the gain 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
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:INPUTMode	Sets or queries the input mode of the probe that is attached to the specified channel
CH<x>:PRObe:INPUTMode:AOFFSet	Sets or queries the requested A mode offset control of the probe that is attached to the specified channel
CH<x>:PRObe:INPUTMode:BOFFSet	Sets or queries the requested B mode offset control of the probe that is attached to the specified channel
CH<x>:PRObe:INPUTMode:CMOFFSet	Sets or queries the requested common mode offset control of the probe that is attached to the specified channel
CH<x>:PRObe:INPUTMode:DMOFFSet	Sets or queries the requested differential mode offset control of the probe that is attached to the specified channel
CH<x>:PRObe:RANge	Sets or queries the attenuation range of the probe on the designated channel
CH<x>:PRObe:RESistance?	Returns the resistance of the probe that is attached to the specified channel
CH<x>:PRObe:SET	Sets or queries aspects of probe accessory user interfaces, for example probe attenuation factors
CH<x>:PRObe:SIGnal	Sets or queries aspects of probe accessory user interfaces
CH<x>:PRObe:TIPtype	Sets or queries the type of probe tip being used on the specified channel
CH<x>:PRObe:UNIts?	Returns the units of the probe that is attached to the specified channel
CH<x>:PROBECal?	Returns the probe calibration status
CH<x>:PROBECOntrol	Sets or queries the probe range
CH<x>:PROBEFunc:EXTAtten	Sets the attenuation value for the specified channel to the specified scale factor Or queries the user-specified attenuation
CH<x>:PROBEFunc:EXTDBatten	Sets the attenuation value for the specified channel to the specified value in decibels Or queries the user-specified attenuation in decibels
CH<x>:PROBEFunc:EXTUnits	Sets the unit of measurement for the external attenuator of the specified channel Or queries the user-specified unit of measurement for the external attenuator
CH<x>:SCAle	Sets or queries the vertical scale of the specified channel
CH<x>:TERmination	Sets channel input termination

Table 2-41: Vertical commands (cont.)

Command	Description
CH<x>:THRESHold	Sets or queries the comparison level for converting the specified channel to digital form
CH<x>:VTERm:BIAS	Sets or queries the termination voltage for a channel (if control is available). <x> is the channel number
CH<x>:VTERm:DUAL:A	Sets or queries the termination voltage for probes with dual inputs that support settable termination voltage
CH<x>:VTERm:DUAL:B	Sets or queries the termination voltage for probes with dual inputs that support settable termination voltage
REF<x>:HORizontal:POSition	Sets or queries 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>:LABel?	Returns a branch query containing the waveform label name and the coordinates at which the label is displayed
REF<x>:LABel:NAMe	Sets or queries the label of the designated waveform
REF<x>:LABel:XPOS	Sets or queries the position of the reference waveform label on the X axis
REF<x>:LABel:YPOS	Sets or queries the position of the reference waveform label on the Y axis
REF<x>:VERTical:POSition	Sets or queries the vertical position of the specified reference waveform
REF<x>:VERTical:SCAle	Sets or queries the reference waveform vertical scale in vertical units/div
SELect?	Returns information on which waveforms are on or off and which waveform is selected
SELect:<wfm>	Turns on the specified waveform Or queries whether the specified channel is on or off
SELect:CH<x>	Sets or queries the displayed state of the specified channel waveform
SELect:CONTROl	Sets or queries the waveform controlled by the front panel

Waveform Transfer command group

Use the commands in the Waveform Transfer Command Group to transfer waveform data points to and from the instrument. 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.

Before you transfer waveform data, you must specify the data format, record length, and waveform source.

Data formats. Acquired waveform data uses eight or more bits to represent each data point. The number of bits used depends on the acquisition mode specified when you acquired the data. Data acquired in SAMple or ENVelope mode uses eight bits per waveform data point. Data acquired in AVERage mode uses up to 14 bits per point.

The instrument can transfer waveform data in either ASCII or binary format. You specify the format with the DATa:ENCdg command. The instrument uses signed, 4 byte integers and floating point values; it does not support unsigned floating point values.

ASCII data is represented by signed integer or floating point values. An example ASCII waveform data string can look like this:

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

Use ASCII to obtain more readable and easier to format output than binary. However, ASCII can require more bytes to send the same values than it does with binary. This can reduce transmission speeds.

Binary data can be represented by signed integer or floating point 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. When a MATH (or REF that came from a MATH) is used, 32-bit floating point values are used that are four bytes in width.

The defined binary formats specify the order in which the bytes are transferred. The following are the four binary formats:

- RIBinary specifies signed integer data-point representation with the most significant byte transferred first.
- 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 PCs.
- RFBinary specifies floating point data-point representation with the most significant byte transferred first.
- SRFBinary is the same as RFBinary 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.

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 instrument, 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 instrument will ignore the value set by DATa:STOP when reading in data. It will stop reading data when there is no more data to read or when it has reached the specified record length.

The CURVe command cannot be used to transfer FastFrame waveforms into the instrument.

When transferring data from the instrument, 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.

For CURVe? queries of FastFrame acquisitions, the data for each frame is concatenated together into one long stream. For example, if you acquire 5 frames of 1000 points each, a curve query (with DATA:START 1 and DATA:STOP 1000) will return 5 * 1000 or 5000 data points. There is no separator between each frame in the data.

If you set DATA:START and DATA:STOP to a subset of the record length then you get that section of data from each frame. For example, with 5 frames of 1000 points each and DATA:START 101 and DATA:STOP 150, you will get points 101-150 for the first frame followed by points 101-150 of the second frame, and similar things. In this case the total number of points returned will be 5 * 50 or 250 points.

Waveform data locations and memory allocation. The DATa:SOUrce command specifies the waveform source when transferring a waveform from the instrument.

Waveforms sent to the instrument are always stored in one of the four reference memory locations. Use the DATa:DESTination command to specify a reference memory location.

Waveform preamble. Each waveform that you transfer has an associated waveform preamble that contains information such as the horizontal scale, the vertical scale, and other settings in effect when the waveform was created. Refer to the individual WFMInpre? and WFMOutpre? commands for more 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 instrument.

You can transfer waveforms from the instrument 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 `WFMOutpre:BYT_Nr`.

NOTE. *MATH waveforms (and REF waveforms that came from a MATH) are always set to four bytes.*

4. Specify the portion of the waveform that you want to transfer using `DATa:STARt` and `DATa:STOP`.
5. Transfer waveform preamble information using `WFMOutpre`.
6. Transfer waveform data from the instrument using `CURVe?`.

Transferring waveform data to the instrument.

1. Specify waveform reference memory using `DATa:DESTination`.
2. Set `WFMInpre:NR_Pt` to equal the number of data points to be sent.
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 the waveform data using `CURVe`.

NOTE. *FastAcq and FastFrame waveforms cannot be returned to the instrument via the CURVe command*

Table 2-42: Waveform Transfer commands

Command	Description
ALLOcate:WAVEform:REF<x>?	Returns the record length for the specified reference waveform, if active, or zero (0) if not active
CURVe	The command format transfers waveform data to the instrument (reference memory location specified by DATa:DESTination) The query format transfers waveform data from instrument specified by the DATa:SOUrce command
CURVENext?	Returns unique waveform data from a new acquisition
CURVEStream	Transfers continuously the waveform data from the instrument as it is acquired
DATa	Sets the format and location of the waveform data that is transferred with the CURVe Command Or queries the format and location of the waveform data that is transferred with the CURVe? command
DATa:DESTination	Sets or queries the reference memory location for storing waveform data sent to the instrument
DATa:ENCdg	Sets or queries the format of outgoing waveform data
DATa:FRAMESTARt	Sets or queries the starting data frame for waveform transfer
DATa:FRAMESTOP	Sets or queries the ending data frame for waveform transfer
DATa:SOUrce	Sets or queries the location of waveform data transferred from the instrument
DATa:STARt	Sets or queries the starting point in waveform transfer
DATa:STOP	Sets or queries the ending data point in waveform transfer
DATa:SYNCSOUrces	Sets or queries if the data sync sources are on or off
WAVFrm?	Returns WFMOutpre? and CURVe? data for the waveform as specified by the DATA:SOUrce command
WAVFRMStream?	Returns WFMOUTPRE? and CURVESTREAM? data for the waveforms specified by the DATASOURCE command
WFMInpre?	Returns the waveform formatting specification to be applied to the next incoming CURVE command data
WFMInpre:BIT_Nr	Sets or queries the number of bits per binary waveform point
WFMInpre:BN_Fmt	Sets or queries the format of binary data for the incoming waveform
WFMInpre:BYT_Nr	Sets or queries the binary field data width for the first ordered waveform
WFMInpre:BYT_Or	Sets or queries the byte order of waveform points
WFMInpre:ENCdg	Sets or queries the type of encoding for incoming waveform data
WFMInpre:NR_Pt	Sets or queries the number of points in the transmitted waveform record
WFMInpre:NR_FR?	This query returns the number of frames to be transferred to the instrument with the Curve command
WFMInpre:PT_Fmt	Sets or queries the point format of incoming waveform data
WFMInpre:PT_Off	Sets or queries the trigger point within the incoming waveform data
WFMInpre:WFId	Allows a waveform extracted from the instrument to be more easily sent back
WFMInpre:XINcr	Sets or queries the horizontal sampling interval between incoming waveform points

Table 2-42: Waveform Transfer commands (cont.)

Command	Description
WFMInpre:XUNit	Sets or queries the horizontal units of the incoming waveform
WFMInpre:XZEro	Sets or queries the (sub-sample) time between the trigger sample and the occurrence of the actual incoming waveform trigger
WFMInpre:YMUlt	Sets or queries the vertical scale factor, per digitizing level, of the incoming waveform points
WFMInpre:YOFf	Sets or queries the vertical position of the incoming waveform in digitizing levels
WFMInpre:YUNit	Sets or queries the vertical units of the incoming waveform
WFMInpre:YZEro	Sets or queries the offset of the incoming waveform
WFMOutpre?	Returns the waveform formatting data for the waveform specified by the DATA:SOURCE command
WFMOutpre:BIT_Nr	Sets or queries the number of bits per waveform point that outgoing waveforms contain
WFMOutpre:BN_Fmt	Sets or queries the format of binary data for the waveform
WFMOutpre:BYT_Nr	Sets or queries the data width for the waveform
WFMOutpre:BYT_Or	Sets or queries the byte order of waveform points
WFMOutpre:ENCdg	Sets or queries the type of encoding for outgoing waveforms
WFMOutpre:NR_Pt?	Returns the number of points for the waveform transmitted in response to a CURVe? query
WFMOutpre:NR_FR?	Query-only command returns the number of frames for the DATa:SOUrce waveform transmitted in response to a CURVe? query
WFMOutpre:PT_Fmt?	Returns the point format for the waveform
WFMOutpre:PT_Off?	Returns the trigger point relative to DATA:START for the waveform
WFMOutpre:PT_ORder?	Returns whether the source waveform is DPO
WFMOutpre:WFId?	Returns a string describing the acquisition parameters for the waveform
WFMOutpre:XINcr?	Returns the horizontal sampling interval
WFMOutpre:XUNit?	Returns the horizontal units for the waveform
WFMOutpre:XZEro?	Returns the (sub-sample) time between the trigger sample and the occurrence of the actual waveform trigger
WFMOutpre:YMUlt?	Returns the vertical scale factor per digitizing level for the waveform
WFMOutpre:YOFf?	Returns the vertical offset in digitizing levels for the waveform
WFMOutpre:YUNit?	Returns the vertical units for the waveform
WFMOutpre:YZEro?	Returns the vertical offset for the waveform

Zoom command group

Zoom commands let you expand and position the waveform display horizontally and vertically, without changing the time base or vertical settings.

You can specify up to four zoom areas. You can also specify a zoomed waveform as a zoom source, which enables you to zoom in on a zoomed waveform.

Table 2-43: Zoom commands

Command	Description
ZOOm	Resets the zoom transforms to default values for all traces or live traces. The ZOOm query returns the current vertical and horizontal positioning and scaling of the display
ZOOm:GRAticule:SIZE	Sets or queries the size of the zoom graticule
ZOOm:HORizontal:POSition	Sets or queries the horizontal position around which the zoom waveform is displayed. It is freely movable within the confines of the acquired waveform (0% to 100%) and measured from left to right of the acquired waveform
ZOOm:HORizontal:SCAle	Sets or queries the zoom horizontal scale factor around which the zoom waveform is displayed
ZOOm:MATH<x>:HORizontal:POSition	Sets or queries the horizontal position of the specified math waveform
ZOOm:MATH<x>:HORizontal:SCAle	Sets or queries the zoom horizontal scale factor of the specified math waveform
ZOOm:MATH<x>:VERTical:POSition	Sets or queries the vertical position of the specified math waveform
ZOOm:MATH<x>:VERTical:SCAle	Sets or queries the vertical scale of the specified math waveform
ZOOm:MODe	Sets or queries the zoom mode
ZOOm:REF<x>:HORizontal:POSition	Sets or queries the horizontal position of the specified reference waveform
ZOOm:REF<x>:HORizontal:SCAle	Sets or queries the zoom horizontal scale factor of the specified reference waveform
ZOOm:REF<x>:VERTical:POSition	Sets or queries the vertical position of the specified reference waveform
ZOOm:REF<x>:VERTical:SCAle	Sets or queries the zoom vertical scale of the specified reference waveform
ZOOm:SCROLL:DIREction	Sets or queries the direction of AutoScroll for zoomed waveforms
ZOOm:SCROLL:LOCk	Sets or queries the state of Scroll Lock, on or off
ZOOm:SCROLL:SPEED	Sets or queries the speed of AutoScroll for zoomed waveforms
ZOOm:STATE	Specifies or returns a trace as zoomed on or off
ZOOm:VERTical:POSition	Sets or queries the vertical position around which the zoom waveform is displayed. It is freely movable within the confines of the acquired waveform (0% to 100%) and measured from left to right of the acquired waveform
ZOOm:VERTical:SCAle	Sets or queries the zoom vertical scale factor around which the zoom waveform is displayed
ZOOm:ZOOM<x>	Sets or queries zoom factors for the specified zoom
ZOOm:ZOOM<x>:DCHAN:DISplay	Sets or queries the display of the digital waveforms for the specified zoom
ZOOm:ZOOM<x>:SCROLLLock	Sets or queries Scroll Lock for the specified zoom
ZOOm:ZOOM<x>:STATE	Sets or queries the specified zoom on or off, where x is an integer from 1 to 4 representing the desired zoom window
ZOOm:ZOOM<x>:<wfm>:DISplay	Sets or queries the display of the specified waveform for the specified zoom

Table 2-43: Zoom commands (cont.)

Command	Description
ZOOm:ZOOM<x>:<wfm>:HORizontal:POSition	Sets or queries the horizontal position of the specified waveform for the specified zoom
ZOOm:ZOOM<x>:<wfm>:HORizontal:SCAle	Sets or queries the horizontal zoom scale of the specified waveform in the specified zoom
ZOOm:ZOOM<x>:<wfm>:VERTical:POSition	Sets or queries the vertical zoom position for the specified waveform in the specified zoom
ZOOm:ZOOM<x>:<wfm>:VERTical:SCAle	Sets or queries the vertical zoom scale for the specified waveform in the specified zoom

Commands listed in alphabetical order

NOTE. *Some of the following commands may not be available on your instrument model. Also, some of the following commands are only available if your instrument has the associated option installed.*

ACQuire:ENHANCEDEnob

This command sets or queries the state of the enhanced effective number of bits.

Group Acquisition

Syntax `ACQuire:ENHANCEDEnob {OFF|AUTO}`

Arguments `OFF` turns off enhanced effective number of bits.

`AUTO` turns enhanced effective number of bits to AUTO.

Examples `ACQUIRE:ENHANCEDENOB AUTO` sets enhanced effective number of bits to AUTO.

ACQuire:ENHANCEDEnob:STATE? (Query Only)

This command queries the state of the enhanced effective number of bits.

Group Acquisition

Syntax `ACQuire:ENHANCEDEnob:STATE?`

Returns `0` indicates that the enhanced effective number of bits in not on.

`1` indicates that the enhanced effective number of bits in on.

Examples `ACQUIRE:ENHANCEDENOB:STATE?` might return `:ACQUIRE:ENHANCEDENOB:STATE 0` indicating that enhanced effective number of bits in not on.

ACQuire:INTERPEightbit

This command sets or queries the interpolation acquisition mode of the instrument.

Group Acquisition

Syntax
```
ACQuire:INTERPEightbit {AUTO|ON|OFF}
ACQuire:INTERPEightbit?
```

Arguments `AUTO` lets the instrument automatically select the interpolation acquisition mode.

`ON` turns on the eight bit interpolation mode.

`OFF` turns off the eight bit interpolation mode.

Examples `ACQUIRE:INTERPEIGHTBIT ON` sets the interpolation sampling mode On.

`ACQUIRE:INTERPEIGHTBIT?` might return `:ACQUIRE:INTERPEIGHTBIT AUTO` indicating the interpolation sampling mode is set to Auto.

ACQuire:MAGnivu

NOTE. *Not available on some models.*

This command sets or queries the MagniVu feature.

This feature provides up to 32 times signal detail for fast viewing of short events and is not recommended for slow data formats such as RS-232.

Group Acquisition

Syntax
```
ACQuire:MAGnivu {<NR1>|OFF|ON}
ACQuire:MAGnivu?
```

Arguments `<NR1>` = 0 disables the MagniVu feature; any other value turns this feature on.

`OFF` disables the MagniVu feature.

`ON` enables the MagniVu feature.

Examples `ACQUIRE:MAGNIVU ON` sets the MagniVu mode to On.

`ACQUIRE:MAGNIVU?` might return `:ACQUIRE:MAGNIVU ON`, indicating that the MagniVu mode is On.

ACQuire:MODe

This command sets or queries the selected acquisition mode of the instrument. This affects all live waveforms. This command is equivalent to selecting Horizontal/Acquisition from the Horiz/Acq menu, and then choosing the desired mode from the Acquisition Mode group box. ACQuire:MODe is the desired acquisition mode.

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 instrument sampling system always samples at the maximum rate and so an acquisition interval can include more 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.

NOTE. *In cases when either **HIRes** or **PkDetect** is the selected mode, and the sampling rate is greater than or equal to the base sampling rate (that is 6.25 Gs/s for 70K oscilloscopes and 5 Gs/s for 5K and 7K oscilloscopes), neither **HIRes** nor **PkDetect** mode is available for the instrument to actually use. Sample mode is used instead, regardless of the mode setting. To determine which mode is actually used in these cases, use the query ACQuire:MODe:ACTUal?*

Group Acquisition

Syntax

```
ACQuire:MODe
{SAMple|PEAKdetect|HIRes|AVErage|WFMDB|ENVelope}
ACQuire:MODe?
```

Related Commands ACQuire:NUMAVg, CURVe, ACQuire:MODe:ACTUal?

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.

`PEAKdetect` specifies the display of high-low range of the samples taken from a single waveform acquisition. The high-low range is displayed as a vertical column that extends from the highest to the lowest value sampled during the acquisition interval. `PEAKdetect` mode can reveal the presence of aliasing or narrow spikes.

`HIRes` specifies Hi Res mode where the displayed data point value is the average of all the samples taken during the acquisition interval. This is a form of averaging, where the average comes from a single waveform acquisition. The

number of samples taken during the acquisition interval determines the number of data values that compose the average.

AVERage specifies averaging mode, in which the resulting waveform shows an average of SAMple data points from several separate waveform acquisitions. The instrument 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:NUMEnv command.

WFMDB (Waveform Database) mode acquires and displays a waveform pixmap. A pixmap is the accumulation of one or more acquisitions.

ENVelope specifies envelope mode, where the resulting waveform shows the PEAKdetect range of data points from several separate waveform acquisitions. The number of waveform acquisitions that go into making up the envelope waveform is set or queried using the ACQuire:NUMEnv command.

The instrument acquires data after each trigger event using Sample mode; it then determines the pix map location of each sample point and accumulates it with stored data from previous acquisitions.

A Pix map is a two dimensional array. The value at each point in the array is a counter that reflects the hit intensity. Infinite and noninfinite persist display modes affect how pix maps are accumulated. Zoom, Math, FastAcq, FastFrame, XY, Roll, and Interpolated Time (IT) Sampling Mode are conflicting features to WFMDB acqMode. Turning on one of them generally turns the other one off. Selection of some standard masks (for example, eye masks, which require option MTM) changes the acquisition mode to WFMDB.

Examples

ACQUIRE:MODE ENVELOPE sets the acquisition mode to display a waveform that is an envelope of many individual 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:MODe:ACTUal? (Query Only)

This query returns the acquisition mode that the instrument is actually using for acquisitions, as opposed to the ACQuire:MODe query, which returns the currently selected acquisition mode. The acquisition mode that the instrument is using may differ from the selected mode in cases where the **HIRes** or **PkDetect** mode has been selected and the sample rate is greater than or equal to the base sampling rate (that is 6.25 Gs/s for 70K oscilloscopes and 5 Gs/s for 5K and 7K oscilloscopes). This is because the **HIRes** and **PkDetect** modes are not available at such high sampling rates. In these cases, the instrument uses sampling mode, and this query returns SAMPLe, regardless of the mode selected.

In other words, when either **HIRes** or **PkDetect** mode has been selected, and the sample rate is greater than or equal to the base sampling rate (that is 6.25 Gs/s for 70K scopes and 5 Gs/s for 5K and 7K oscilloscopes), use the ACQuire:MODe query to determine the current mode selection; use `ACQuire:MODe:ACTUal?` to determine which mode the instrument is actually using. For other modes, use either query, as they will return the same mode.

Group Acquisition

Syntax `ACQuire:MODe:ACTUal?`

Related Commands ACQuire:MODe

Examples `ACQUIRE:MODE:ACTUAL?` might return the following:

In cases where the sampling rate ≥6.25 Gs/s for 70K oscilloscopes and 5 Gs/s for 5K and 7K oscilloscopes, and **HIRes** or **PkDetect** is the currently selected mode, then `ACQuire:MODe:ACTUal?` will return `SAMPLE`.

In cases where the sampling rate < 6.25 Gs/s for 70K oscilloscopes and 5 Gs/s for 5K and 7K oscilloscopes, and **HIRes** or **PkDetect** is the currently selected mode, then `ACQuire:MODe:ACTUal?` will return either `HI RESOLUTION` or `PEAK DETECT`.

In cases where one of the other modes has been selected (that is `AVErage`, `WFMDB` or `ENVelope`), then `ACQuire:MODe:ACTUal?` will return `AVERAGE`, `WFMDB` or `ENVELOPE`.

ACQuire:NUMACq? (Query Only)

This query-only command returns the number of waveform acquisitions that have occurred since starting acquisition with the ACQuire:STATE RUN command. This value is reset to zero when any acquisition, horizontal, or vertical arguments that affect the waveform are changed. The maximum number of acquisitions that can be counted is 2^{30}–1. The instrument stops counting when this number is reached. This is the same value that displays in the lower right of the screen.

Group Acquisition

Syntax `ACQuire:NUMACq?`

Related Commands ACQuire:STATE

Examples `ACQUIRE:NUMACQ?` might return `:ACQUIRE:NUMACQ 350`, indicating that 350 acquisitions have occurred since executing an ACQuire:STATE RUN command.

ACQuire:NUMAVg

This command sets or queries 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 selecting Horizontal/Acquisition Setup from the Horiz/Acq menu, selecting the Acquisition tab, and choosing Average from the Acquisition Mode group box. Then enter the desired number of waveforms that will be used to make up an averaged waveform in the # of Wfms box.

Group Acquisition

Syntax

```
ACQuire:NUMAVg <NR1>
ACQuire:NUMAVg?
```

Related Commands ACQuire:MODe

Arguments `<NR1>` is the number of waveform acquisitions to average.

Examples `ACQUIRE:NUMAVG 10` specifies that 10 waveform averages will be performed before exponential averaging starts.

`ACQUIRE:NUMAVG?` might return `:ACQUIRE:NUMAVG 75`, indicating that there are 75 acquisitions specified for averaging.

ACQuire:NUMEnv

This command sets or queries the number of waveform acquisitions that make up an envelope waveform. Sending this command is equivalent to setting the # of Wfms in the Acquisition Setup menu when Envelope Acquisition mode is selected.

NOTE. *Sending INFInite as an argument sets 2e9 envelopes. The query response will be 2e9 not INFInite.*

Group Acquisition

Syntax

```
ACQuire:NUMEnv {<NR1>|INFInite}
ACQuire:NUMEnv?
```

Arguments

<NR1> is the number of waveform acquisitions to be enveloped.

INFInite specifies the number of waveform acquisitions to be enveloped. Setting the argument to infinite sets the envelopes to 2e9.

Examples

ACQUIRE:NUMENV 10 specifies that an enveloped waveform will show the result of combining 10 separately acquired waveforms.

ACQUIRE:NUMENV? might return :ACQUIRE:NUMENV 75, indicating that 75 acquisitions are acquired for enveloped waveforms.

ACQuire:NUMFRAMESACQuired? (Query Only)

This query-only command returns the number of frames acquired when in FastFrame Single Sequence and acquisitions are running. When the FastFrame single sequence stop condition is set to Stop Manually, this number counts past the number of frames which were requested as new frames are acquired, until STOP is pushed. When not in FastFrame Single Sequence, this value is the same number reported by ACQ:NUMACQ.

Group Acquisition

Syntax

```
ACQuire:NUMFRAMESACQuired?
```

Related Commands ACQuire:NUMACq?

Examples

ACQUIRE:NUMFRAMESACQUIRED? might return :ACQUIRE:NUMFRAMESACQUIRED 350, indicating that 350 frames have been acquired.

ACQuire:NUMSAMples

This command sets or queries the minimum number of acquired samples that make up a waveform database (WfmDB) waveform for single sequence mode and Mask Pass/Fail Completion Test. This is equivalent to setting the Waveform Database Samples in the Acquisition Mode side menu.

Group Acquisition

Syntax

```
ACQuire:NUMSAMples <NR1>
ACQuire:NUMSAMples?
```

Arguments

<NR1> is the minimum number of acquired samples that make up a waveform database (WfmDB) waveform for single sequence mode and Mask Pass/Fail Completion Test. The default value is 16,000 samples. The range is 5,000 to 2,147,400,000 samples.

In real time mode, if the entire trace is displayed, one triggered acquisition should acquire 5000 samples for a record length of 5000. If you want 5001 samples, the instrument will need to make two acquisitions (10,000 samples). If you request fewer samples than the record length, the number of samples will be the same as the record length.

Examples

ACQUIRE:NUMSAMPLES 5000 specifies that 5000 samples will be acquired in WfmDB mode for single sequence mode and Mask Pass/Fail Completion Test.

ACQUIRE:NUMSAMPLES? might return :ACQUIRE:NUMSAMples 10000, indicating that there are 10,000 samples to be acquired in WfmDB mode for single sequence and Mask Pass/Fail Completion Test.

ACQuire:SAMPlingmode

This command sets or queries the sampling mode.

Group

Acquisition

Syntax

```
ACQuire:SAMPlingmode {RT|ET|IT}
ACQuire:SAMPlingmode?
```

Arguments

RT sets the sampling mode to real time only.

ET sets the sampling mode to equivalent time allowed (ON in REPET).

IT sets the sampling mode to interpolation allowed (OFF in REPET).

The number of channels selected and the sampling speed affects the mode that the instrument uses to acquire data. At slow sampling speeds the instrument uses real time sampling. At fast sampling speeds, the instrument uses equivalent time sampling or interpolation. For speeds between these extremes, some instruments create waveform records differently, depending on the number of input channels selected.

Examples

ACQUIRE:SAMPLINGMODE IT specifies that interpolation is allowed.

ACQUIRE:SAMPLINGMODE? might return :ACQUIRE:SAMPLINGMODE RT, indicating that sampling mode is real time only.

ACQuire:STATE

This command 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 instrument resets the number of acquisitions. If the RUN argument is issued while in continuous mode, a reset occurs and acquired data continues to acquire.

If acquire:stopafter is SEQUENCE, pressing the front panel RUN/STOP button is equivalent to sending these commands: ACQUIRE:STOPAFTER RUNSTOP and either ACQUIRE:STATE ON or ACQUIRE:STATE RUN.

Group

Acquisition

Syntax

```
ACQuire:STATE {<NR1>|OFF|ON|RUN|STOP}
ACQuire:STATE?
```

Related Commands

ACQuire:STOPAfter

Arguments

<NR1> = 0 stops acquisitions; any other value starts acquisitions.

OFF stops acquisitions.

ON starts acquisitions.

RUN starts acquisitions.

STOP stops acquisitions.

Examples

ACQUIRE:STATE RUN starts the acquisition of waveform data and resets the count of the number of acquisitions.

ACQUIRE:STATE? might return :ACQUIRE:STATE 0, indicating that the acquisition is stopped.

ACQuire:STOPAfter

This command sets or queries whether the instrument continually acquires acquisitions or acquires a single sequence. Pressing SINGLE on the front panel button is equivalent to sending these commands: `ACQUIRE:STOPAFTER SEQUENCE` and `ACQUIRE:STATE 1`.

Group Acquisition

Syntax

```
ACQuire:STOPAfter {RUNSTop|SEQuence}
ACQuire:STOPAfter?
```

Related Commands ACQuire:STATE

Arguments

`RUNSTop` specifies that the instrument will continually acquire data, if `ACQuire:STATE` is turned on.

`SEQuence` specifies that the next acquisition will be a single-sequence acquisition.

Examples

`ACQUIRE:STOPAFTER RUNSTOP` sets the instrument to continually acquire data.

`ACQUIRE:STOPAFTER?` might return `:ACQUIRE:STOPAFTER SEQUENCE`, indicating that the next acquisition the instrument makes will be of the single-sequence type.

ALIas

This command sets or queries the state of alias functionality.

Group Alias

Syntax

```
ALIas {OFF|ON|<NR1>}
ALIas?
```

Related Commands ALIas:STATE

Arguments

`OFF` turns Alias expansion off. If a defined alias is sent when `ALIas:STATE` is off, a command error (102) will be generated.

`ON` turns Alias expansion on. When a defined alias is received, the specified command sequence is substituted for the alias and executed.

<NR1> = 0 disables Alias mode; any other value enables Alias mode.

Examples

ALIAS ON turns the alias feature on. When a defined alias is received, the specified command sequence is substituted for the alias and executed.

ALIAS? might return :ALIAS:STATE 1, indicating that the alias feature is on.

ALIas:CATalog? (Query Only)

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

Group

Alias

Syntax

```
ALIas:CATalog?
```

Examples

ALIAS:CATALOG? might return the string :ALIAS:CATALOG "SETUP1","TESTMENU1","DEFAULT" showing that there are three aliases named SETUP1, TESTMENU1, and DEFAULT.

ALIas:DEFine

This command 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 is 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.*

Group

Alias

Syntax

```
ALIas:DEFine <QString><,>{<QString>|<Block>}
ALIas:DEFine?
```

Related Commands

ALIas:STATE

Arguments The first <QString> is the alias label.

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.

The second <QString> or <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 256 characters.

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

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

ALIas:DELETe (No Query Form)

This command 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)

This command 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)

This command removes a specified alias. An error message is generated if the named alias does not exist. 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 exception error.`<QString>` must be a previously defined alias.

Examples `ALIAS:DELETE:NAME "STARTUP"` deletes the alias named STARTUP.

ALIas:STATE

This command turns aliases on or off and is identical to the ALIas command. The query form of this command returns the state of the aliases.

Group Alias

Syntax
```
ALIas:STATE {<NR1>|OFF|ON}
ALIas:STATE?
```

Arguments `<NR1>` = 0 turns off aliases; any other value turns on aliases.

`OFF` turns alias expansion off. If a defined alias is sent when `ALIas:STATE` is `OFF`, the instrument generates a command error (102).

`ON` 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 off the alias feature.

`ALIAS:STATE?` might return `:ALIAS:STATE ON`, indicating that alias expansion is currently turned on.

ALLEv? (Query Only)

This query-only command prompts the instrument 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 instrument.

Group Status and Error

Syntax `ALLEv?`

Related Commands *ESR?, EVMsg?

Examples `ALLEV?` might return `:ALLEV 2225,"Measurement error, No waveform to measure; "420,"Query UNTERMINATED;"`.

ALLOcate:WAVEform:REF<x>? (Query Only)

This query-only command returns the record length for the specified reference waveform, if active, or zero (0) if not active (that is, no slot exists for the reference waveform).

Group Waveform Transfer

Syntax `ALLOcate:WAVEform:REF<x>?`

Examples `ALLOCATE:WAVEFORM:REF3?` might return 5000 if a 5000 point waveform was active in the reference 3 slot.

APPLication:ACTivate (No Query Form)

For legacy Java based applications, starts the application specified in the string. The available applications depend on the oscilloscope model and installed options. (DPOJET and DPOJET-based applications do not require this command

and are started automatically by sending them any command, such as DPOJET:VERsion?)

Group Miscellaneous

Syntax APPLication:ACTivate <string>

Related Commands APPLication:SCOPEAPP:WINDOW

Arguments <string> is the name of the application that you want to start. Enter the application name exactly as it appears in the oscilloscope Analyze menu.

Examples APPLICATION:ACTIVATE "PCI Express" starts the PCI Express application.

APPLication:SCOPEAPP:WINDOW

Sets or queries whether the oscilloscope application is minimized or displayed full screen.

Group Miscellaneous

Syntax
```
APPLication:SCOPEAPP:WINDOW {FULLSCREEN|MINIMIZED}
APPLication:SCOPEAPP:WINDOW?
```

Related Commands APPLication:ACTivate

Arguments FULLSCREEN sets the oscilloscope display to fill the oscilloscope screen.

MINIMIZED minimizes the oscilloscope display to an icon at the bottom of the screen.

Examples APPLICATION:SCOPEAPP:WINDOW FULLSCREEN causes the oscilloscope display to fill the oscilloscope screen.

APPLICATION:SCOPEAPP:WINDOW? might return :APPLICATION:SCOPEAPP:WINDOW MINIMIZED, indicating that the oscilloscope display has been minimized.

AUTOSet (No Query Form)

This command (no query format) sets the vertical, horizontal, and trigger controls of the instrument to automatically acquire and display the selected waveform. (To autoset a video waveform, the video trigger must be set to video standard, not custom. Video arguments require video hardware.) This is equivalent to pressing the front panel AUTOSET button. For a detailed description of autoset functionality, see *Autoset* in the index of the online help for your instrument.

Group Miscellaneous

Syntax

```
AUTOSet {EXECute|UNDo|VFields|VIDeo|VLines}
```

Related Commands DISplay:GRAticule

Arguments

`EXECute` autosets the displayed waveform; this is equivalent to pressing the front panel AUTOSET button. If the display is set to a PAL, MV, or IRE graticule, this argument forces the graticule display to full mode (frame, grid, and cross hair).

`UNDo` returns the instrument to the setting prior to executing an autoset.

`VFields` autosets the displayed waveform.

`VIDeo` autosets the displayed waveform.

`VLines` autosets the displayed waveform.

Examples `AUTOSET VFields` sets the instrument to trigger on all video fields.

AUXIn:BANdwidth

This command sets or queries the selectable low-pass bandwidth limit filter of the AUX In input. The bandwidth will change with an appropriate probe attached. This is equivalent to selecting Bandwidth from the Vertical menu.

The query form of this command always returns the maximum bandwidth of the AUX In input.

Group Vertical

Syntax

```
AUXIn:BANdwidth <NR3>
AUXIn:BANdwidth?
```

Arguments <NR3> is one of the supported bandwidths on the attached probe.

Examples AUXIN:BANDWIDTH 10.0000E+9 sets the bandwidth of AUX In input to 10 GHz.

AUXIN:BANDWIDTH?, for DPO7054 instruments, might return :AUXIN:BANDWIDTH 500.0000E+06, indicating that there is no bandwidth limiting on the AUX In input.

AUXIn:COUPling

This command sets or queries the input attenuator coupling setting for the AUX In input. The coupling will change with an appropriate probe attached. This command is equivalent to selecting Coupling from the Vertical menu.

Group Vertical

Syntax
```
AUXIn:COUPling <NR3>
AUXIn:COUPling?
```

Arguments <NR3> is one of the supported input attenuator couplings on the attached probe.

Examples AUXIN:COUPLING GND sets the AUX In input to ground.

AUXIN:COUPling? might return :AUXIN:COUPling GND, indicating that AUX In input is set to GND coupling.

AUXIn:OFFSet

This command sets or queries the vertical offset for the AUX In input. This command is equivalent to selecting Offset from the Vertical menu.

This command offsets the vertical acquisition window (moves the level at the vertical center of the acquisition window) for the input. 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 to help determine what data is acquired. The instrument always displays the input signal minus the offset value. The channel reference marker will move to the vertical graticule position given by the negative of the offset value divided by the scale factor, unless that position is off-screen. If the computed coordinate for the reference

mark is off-screen, the mark moves to the nearest screen limit and changes from a right-pointing arrow (→) to an arrow pointing in the appropriate off-screen direction.

Group Vertical

Syntax
```
AUXIn:OFFSet <NR3>
AUXIn:OFFSet?
```

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

Examples AUXIN:OFFSet 2.0E-3 sets the offset for AUX In input to 2 mV.

AUXIN:OFFSet? might return :AUXIN:OFFSET 1.0000E-03, indicating that the offset for AUX In input is set to 1 mV.

AUXIn:PRObe:AUTOZero (No Query Form)

The command sets the probe attached to the AUX In input to autozero.

Group Vertical

Syntax
```
AUXIn:PRObe:AUTOZero EXECute
```

Arguments EXECute sets the probe to autozero.

Examples AUXIN:PROBE:AUTOZERO EXECUTE sets the probe to autozero.

AUXIn:PRObe:DEGAUSS (No Query Form)

This command starts a degauss cycle of the probe attached to the AUX In input. The degauss cycle will change with an appropriate probe attached.

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)

This command queries whether the probe attached to the AUX In input is degaussed.

Group Vertical

Syntax AUXIn:PRObe:DEGAUSS:STATE?

Examples AUXIN:PROBE:DEGAUSS:STATE? might return AUXIN:PROBE:DEGAUSS:STATE PASSED, indicating that the probe is degaussed.

AUXIn:PRObe:FORCEDRange

This command sets the probe attached to the AUX In input to the specified range, or the command queries the probe range.

Group Vertical

Syntax
```
AUXIn:PRObe:FORCEDRange <NR3>
AUXIn:PRObe:FORCEDRange?
```

Arguments <NR3> specifies the probe range.

Examples If a TCP0030 current probe is attached the AUX IN input, AUXIN:PROBE:FORCEDRANGE 5.0 sets the probe to its 5 Ampere range.

AUXIN:PROBE:FORCEDRANGE might return :AUXIN:PROBE:FORCEDRANGE 10.0000E+36, indicating that the probe range is set to 10.0000E+36 dB.

AUXIn:PRObe:GAIN? (Query Only)

This query-only command returns the gain factor of the probe that is attached to the AUX In input. 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.

Group Vertical

Syntax `AUXIn:PRObe:GAIN?`

Examples `AUXIN:PROBE:GAIN?` might return `:AUXIN:PROBE:GAIN 0.1000E+00`, 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)

This query-only command returns the probe serial number.

Group Vertical

Syntax `AUXIn:PRObe:ID:SERnumber?`

Examples `AUXIN:PROBE:ID:SERNUMBER?` might return `:AUXIN:PROBE:ID:SERNUMBER "B999999"`, indicating that the probe serial number is set to B999999.

AUXIn:PRObe:ID:TYPe? (Query Only)

This query-only command returns the probe type.

Group Vertical

Syntax `AUXIn:PRObe:ID:TYPe?`

Examples `AUXIN:PROBE:ID:TYPE?` might return `:AUXIN:PROBE:ID:TYPE "TAP1500"`, indicating that the probe type is set to TAP1500.

AUXIn:PRObe:INPUTMode

This command sets or queries one of the probe's four input modes on the AUX In input.

Group Vertical

Syntax

```
AUXIn:PRObe:INPUTMode {COMmonmode|DEFault|DIFFerential|A|B}
AUXIn:PRObe:INPUTMode?
```

Arguments

`COMmonmode` sets the probe to route common-mode signals to the host.

`DEFault` sets the probe to the default mode.

`DIFFerential` sets the probe to route differential signals to the host.

`A` sets the probe to route single-ended A signals to the host.

`B` sets the probe to route single-ended B signals to the host.

Examples

`AUXIN:PROBE:INPUTMODE A` sets the probe to route single-ended A signals to the host.

`AUXIN:PROBE:INPUTMODE?` might return `:AUXIN:PROBE:INPUTMODE DEFAULT`, indicating that the probe input mode on the AUXIN input is set to Default.

AUXIn:PRObe:INPUTMode:AOFFSet

This command sets or queries the requested A mode offset control of the probe that is attached to the AUX In input.

Group Vertical

Syntax

```
AUXIn:PRObe:INPUTMode:AOFFSet <NR3>
AUXIn:PRObe:INPUTMode:AOFFSet?
```

Arguments

`<NR3>` specifies the A mode offset control.

Examples

`AUXIN:PROBE:INPUTMODE:AOFFSET 0.2` sets the A mode offset control to 0.2 V.

AUXIN:PROBE:INPUTMODE:AOFFSET? might return :AUXIN:PROBE:INPUTMODE:AOFFSET 0.0000, indicating that the A mode offset control is set to 0 V.

AUXIn:PRObe:INPUTMode:BOFFSet

This command sets or queries the requested B mode offset control of the probe that is attached to the AUX In input.

Group Vertical

Syntax

```
AUXIn:PRObe:INPUTMode:BOFFSet <NR3>
AUXIn:PRObe:INPUTMode:BOFFSet?
```

Arguments <NR3> specifies the B mode offset control.

Examples AUXIN:PROBE:INPUTMODE:BOFFSET 0.2 sets the B mode offset control to 0.2 V.

AUXIN:PROBE:INPUTMODE:BOFFSET? might return :AUXIN:PROBE:INPUTMODE:BOFFSET? “0.0000”, indicating that the B mode offset control is set to 0 V.

AUXIn:PRObe:INPUTMode:CMOFFSet

This command sets or queries the requested common mode offset control of the probe that is attached to the AUX In input.

Group Vertical

Syntax

```
AUXIn:PRObe:INPUTMode:CMOFFSet <NR3>
AUXIn:PRObe:INPUTMode:CMOFFSet?
```

Arguments <NR3> specifies the common mode offset control

Examples AUXIN:PROBE:INPUTMODE:CMOFFSET 0.2 sets the common mode offset control to 0.2 V.

AUXIN:PROBE:INPUTMODE:CMOFFSET? might return :AUXIN:PROBE:INPUTMODE:CMOFFSET? "0.0000", indicating that the common mode offset control is set to 0 V.

AUXIn:PRObe:INPUTMode:DMOFFSet

This command sets or queries the requested differential mode offset control of the probe that is attached to the AUX In input.

Group Vertical

Syntax

```
AUXIn:PRObe:INPUTMode:DMOFFSet <NR3>
AUXIn:PRObe:INPUTMode:DMOFFSet?
```

Arguments <NR3> specifies the differential mode offset control.

Examples AUXIN:PROBE:INPUTMODE:DMOFFSET 0.2 sets the differential mode offset control to 0.2 V.

AUXIN:PROBE:INPUTMODE:DMOFFSET? might return :AUXIN:PROBE:INPUTMODE:DMOFFSET? "0.0000", indicating that the differential mode offset control is set to 0 V.

AUXIn:PRObe:RANge

This command controls or queries the set attenuation range of the probe on the AUX In input.

Group Vertical

Syntax

```
AUXIn:PRObe:RANge
{ATTEN1X|ATTEN1_25X|ATTEN1_5X|ATTEN1_75X|ATTEN2X|ATTEN2_5X|
ATTEN3X|ATTEN3_5X|ATTEN4X|ATTEN4_5X|ATTEN5X|ATTEN5_5X|
ATTEN6X|ATTEN6_5X|ATTEN7X|ATTEN7_5X|ATTEN8X|ATTEN8_5X|
ATTEN9X|ATTEN9_5X|ATTEN10X|ATTEN12_5X|ATTEN15X|ATTEN17_5X|
ATTEN20X|ATTEN25X|ATTEN30X|ATTEN35X|ATTEN40X|ATTEN45X|
ATTEN50X|ATTEN55X|ATTEN60X|ATTEN65X|ATTEN70X|ATTEN75X|
ATTEN80X|ATTEN85X|ATTEN90X|ATTEN95X|ATTEN100X|ATTEN125X|
ATTEN150X|ATTEN175X|ATTEN200X|ATTEN250X|ATTEN300X|
ATTEN350X|ATTEN400X|ATTEN450X|ATTEN500X|ATTEN550X|
ATTEN600X|ATTEN650X|ATTEN700X|ATTEN750X|ATTEN800X|
ATTEN850X|ATTEN900X|ATTEN950X|ATTEN1000X}
AUXIn:PRObe:RANge?
```

Examples AUXIN:PROBE:RANGE ATTEN10X sets the probe attenuation to 10X.

AUXIN:PROBE:RANGE? might return :AUXIN:PROBE:RANGE ATTEN1_25X, indicating that the probe attenuation is set to 25X.

AUXIn:PRObe:RESistance? (Query Only)

This query-only command returns the resistance factor of the probe that is attached to the AUX In input.

Group Vertical

Syntax AUXIn:PRObe:RESistance?

Examples AUXIN:PROBE:RESISTANCE? might return :AUXIN:PROBE:RESISTANCE 10.0000E+06, indicating that the input resistance of the probe attached to the AUXIN input is set to 10 MΩ.

AUXIn:PRObe:SET

This command sets or queries aspects of probe accessory user interfaces, for example probe attenuation factors. The available arguments for this command will vary depending on the accessory you attach to the instrument auxiliary trigger input.

For the P7260 probe, you can select between two attenuation factors using either this GPIB command or the push switch on the probe. The probe enables the relevant path and adjusts the settings based on the characteristics of the path in use. The probe signal path selection is not kept in persistent storage. The probe will lose the selection if you reboot the instrument or remove the probe. Also, the instrument does not store the selection in the save/recall setup operation.

Group Vertical

Syntax
```
AUXIn:PRObe:SET <QString>
AUXIn:PRObe:SET?
```

Arguments <QString> is a quoted string representing a settable aspect of the attached accessory.

Example strings for some probes are shown as follows:

ATTENUATION 5X sets the P7260 probe to ±0.75 V dynamic range with 6 GHz bandwidth and 5X attenuation.

ATTENUATION 25X sets the P7260 probe to ±3 V dynamic range with 6 GHz bandwidth and 25X attenuation.

VTERMsource AUTO sets the P7380SMA probe voltage termination source to auto.

VTERMsource INTernal sets the P7380SMA probe voltage termination source to internal.

VTERMsource EXTernal sets the P7380SMA probe voltage termination source to external.

Examples

AUXIN:PROBE:SET "ATTEN 25X" sets the P7260 probe to the selection for ±3 V dynamic range with 3 GHz bandwidth and 25X attenuation.

AUXIN:PROBE:SET? might return :CH2:PROBE:SET "ATTENUATION 5X" indicating that the P7260 probe is set to the selection for ±0.75 V dynamic range with 6 GHz bandwidth and 5X attenuation.

AUXIN:PROBE:SET? "ATTEN" might return :CH2:PROBE:SET "ATTENUATION 25X", indicating that the P7260 probe is set to the selection for ±3 V dynamic range with 3 GHz bandwidth and 25X attenuation.

AUXIn:PRObe:SIGnal

This command sets or queries aspects of probe accessory user interfaces. The available arguments for this command will vary depending on the accessory you attach to the instrument.

Group

Vertical

Syntax

```
AUXIn:PRObe:SIGnal {PASS|BYPass}
AUXIn:PRObe:SIGnal?
```

Arguments

PASS opens a relay passing your signal to the instrument.

BYPass closes a relay preventing your signal from reaching the instrument. During probe degauss, the signal should be bypassed.

Examples

AUXIN:PROBE:SIGNAL PASS opens a relay passing your signal to the instrument.

AUXIN:PROBE:SIGNAL? might return :AUXIN:PROBE:SIGNAL BYPASS, indicating that the relay is closed preventing your signal from reaching the instrument.

AUXIn:PRObe:TIPtype

This command sets or queries the type of probe tip being used.

Group Vertical

Syntax

```
AUXIn:PRObe:TIPtype {HBWStraightflex|OTHer}
AUXIn:PRObe:TIPtype?
```

Arguments

`HBWStraightflex` lets the instrument know you are using a high bandwidth straight-flex probe tip.

`OTHer` lets the instrument know you are not using a high bandwidth straight-flex probe tip.

Examples

`AUXIN:PROBE:TIPTYPE HBWStraightflex` sets the auxiliary input probe tip type to HBWStraightflex.

`AUXIN:PROBE:TIPTYPE?` might return `:AUXIN:PROBE:TIPTYPE OTHER`, indicating the probe tip is not a HBWStraightflex type.

AUXIn:PRObe:UNIts? (Query Only)

This query-only command returns a string describing the units of measure for 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.

AUXIn:PROBEFunc:EXTAtten

This command sets the input-output ratio, of external attenuation or gain, between the signal and the AUX In input.

The query form of this command returns the user-specified attenuation. Note that this command deals with an attenuation factor, not a gain factor, unlike CH<x>:PRObe? (This command returns a value independent of the external attenuation). For example, if you specify a 20X attenuation factor, the commands

return the following values (assuming that a 1x probe is presently attached, since the external attenuation is used in combination with the probe attenuation):

```
AUXIN:PROBE:EXTA? 20.00E+0
```

```
AUXIN:PROBE? 1.0E+0
```

This command is equivalent to selecting Attenuation from the Vertical menu, and then either viewing or setting Ext Atten.

Group Vertical

Syntax

```
AUXIn:PROBEFunc:EXTAtten <NR3>
AUXIn:PROBEFunc:EXTAtten?
```

Related Commands AUXIn:PROBEFunc:EXTDBatten

Arguments <NR3> is the attenuation value, which is specified as a multiplier in the range from 1.00E-10 to 1.00E+10.

Examples AUXIN:PROBEFUNC:EXTATTEN 167.00E-3 sets specifies an external attenuation, which is connected between the user's input signal and the input of the probe attached to the AUX In input.

AUXIN:PROBEFUNC:EXTATTEN? might return :AUXIN:PROBEFUNC:EXTATTEN 1.0000E+00, indicating that the attached probe is connected directly to the user's signal.

AUXIn:PROBEFunc:EXTDBatten

This command sets the input-output ratio (expressed in decibel units) of external attenuation or gain between the signal and the instrument AUX In input.

The query form of this command returns the user-specified attenuation in decibels: 1X = 0 dB, 10X = 20 dB, 100X = 40 dB, and similar things.

This command is equivalent to selecting Attenuation from the Vertical menu, and then either viewing or setting Ext Att(dB).

Group Vertical

Syntax

```
AUXIn:PROBEFunc:EXTDBatten <NR3>
AUXIn:PROBEFunc:EXTDBatten?
```

Related Commands AUXIn:PROBEFunc:EXTDBatten

Arguments <NR3> is the attenuation value, which is specified in the range from -200.00 dB to 200.00 dB.

Examples AUXIN:PROBEFUNC:EXTDBATTEN 2.5 sets the attenuation value on the AUX IN input to 2.5 dB.

AUXIN:PROBEFUNC:EXTDBATTEN? might return :AUXIN:PROBEFUNC:EXTDBATTEEN 2.5000E+00, indicating that the attenuation value on the AUX IN input is set to 2.5 dB.

AUXIn:PROBEFunc:EXTUnits

This command sets the unit of measurement for the external attenuator of the AUX In input. There is also a corresponding query that returns the user-specified unit of measurement for the external attenuator. Unless these units are set to the factory default string value of "None", they become the attenuated units of measurement for the input. It is assumed that the probe connected to the input is of the correct type to receive the output of the user's external transducer or network.

Group Vertical

Syntax
```
AUXIn:PROBEFunc:EXTUnits <QString>
AUXIn:PROBEFunc:EXTUnits?
```

Related Commands AUXIn:PRObe:UNIts?

Arguments <QString> can contain a string of up to eight characters to indicate the attenuation unit of measurement for the AUX In input. However, most instrument attenuators only display the first two characters.

Examples AUXIN:PROBEFUNC:EXTUNITS "Pa" sets the unit of measurement for the AUX IN external attenuator.

AUXIN:PROBEFUNC:EXTUNITS? might return :AUXIN:PROBEFUNC:EXTUNITS "Pa", indicating that the AUX IN external attenuator units of measurement are Pascals.

AUXIn:VTERm:DUAL:A

This command sets or queries the termination voltage for probes with dual inputs that support settable termination voltage.

Group Vertical

Syntax
```
AUXIn:VTERm:DUAL:A <NR3>
AUXIn:VTERm:DUAL:A?
```

Arguments <NR3> specifies the termination voltage.

Examples AUXIN:VTERM:DUAL:A 1.5 sets the termination voltage of the A input of the AUXIN probe to 1.5 Volts.

AUXIN:VTERm:DUAL:A? might return AUXIN:VTERM:DUAL:A 0.0000, indicating that the termination voltage for the A input of the probe plugged into the AUX In input is set to 0 volts.

AUXIn:VTERm:DUAL:B

This command sets or queries the termination voltage for probes with dual inputs that support settable termination voltage.

Group Vertical

Syntax
```
AUXIn:VTERm:DUAL:B <NR3>
AUXIn:VTERm:DUAL:B?
```

Arguments <NR3> specifies the termination voltage.

Examples AUXIN:VTERM:DUAL:B 0.5 sets the termination voltage of the B input of the AUXIN probe to 0.5 Volts.

AUXIN:VTERm:DUAL:B? might return AUXIN:VTERM:DUAL:B 0.0000, indicating that the termination voltage for the B input of the probe plugged into the AUX In input is set to 0 volts.

AUXout? (Query Only)

This query-only command returns the auxiliary output setup and is equivalent to selecting External Signals. From the Utilities menu, and then viewing the current settings for the AUX OUT Configuration.

Group Miscellaneous

Syntax

```
AUXout?
```

Related Commands AUXout:SOUrce

Examples `AUXOUT?` might return `:AUXOUT:SOURCE ATRIGGER;EDGE RISING`, indicating that the source at the BNC connector is set to the A trigger and the polarity is set to the rising edge of the trigger output signal.

AUXout:EDGE

This command sets or queries the direction in which the trigger output signal will transition when a trigger occurs.

Group Miscellaneous

Syntax

```
AUXout:EDGE {RISing|FALling}
AUXout:EDGE?
```

Arguments `RISing` sets the polarity to the rising edge.

`FALling` sets the polarity to the falling edge.

Examples `AUXOUT:EDGE RISING` sets the polarity to rising edge.

`AUXOUT:EDGE?` might return `:AUXOUT:EDGE FALLING`, indicating that the polarity is set to falling edge.

AUXout:SOUrce

This command sets or queries the trigger source at the BNC connection. This command is equivalent to selecting External Signals from the Utilities menu and then selecting the desired Configuration setting.

Group Miscellaneous

Syntax
```
AUXout:SOUrce {ATRIGger|BTRIGger|DELayed|EVENT|REFOUT}
AUXout:SOUrce?
```

Related Commands AUXout?

Arguments `ATRIGger` sets the source at the BNC connector to the main trigger.

`BTRIGger` sets the source at the BNC connector to the delayed trigger. (7K/70K)

`EVENT` sets the source at the BNC connector to a specified event. (7K/70K)

`DELayed` sets the source at the BNC connector to the delayed trigger.

`REFOUT` sets the source at the BNC connector to the reference output. (5K/7K)

Examples `AUXOUT:SOURCE?` might return `:AUXOUT:SOURCE ATRIGGER`, indicating that the source at the BNC connector is set to the A trigger.

BELl (No Query Form)

This command was previously used to beep an audio indicator and is provided for backward compatibility.

Group Miscellaneous

Syntax
```
BELl
```

Examples `BELL` is accepted but does nothing.

BUS:B1<x>:DISplay:HIERarchical

This command sets or queries the display of a bus layer on or off.

Group Bus

Syntax
```
BUS:B1<x>:DISplay:HIERarchical <QString>
BUS:B1<x>:DISplay:HIERarchical?
```

Arguments <QString> consists of two items; the layer identifier and the ON/OFF value.

Examples BUS:B1:DISplay:HIERarchical "BUS1.Lane0 OFF" sets the display of bus 1, lane 0 off.

BUS:B1<x>:DISplay:LAYout

This command sets or queries the format a bus layer should use.

Group Bus

Syntax
```
BUS:B1<x>:DISplay:LAYout <QString>
BUS:B1<x>:DISplay:LAYout?
```

Arguments <QString> consists of two items; the layer identifier and the format identifier.

Examples BUS:B1:DISplay:LAYOUT "BUS1.Lane0 Hex" sets the display layout to display bus 1, lane 0 in hex format.

BUS:B1<x>:USB:HYSTeresis

This command sets or queries the hysteresis for USB Super Speed.

Group Bus

Syntax
```
BUS:B1<x>:USB:HYSTeresis <NR3>
BUS:B1<x>:USB:HYSTeresis?
```

Arguments <NR3> sets the hysteresis for USB Super Speed.

Examples BUS:B1:USB:HYSTeresis? might return :BUS:B1:USB:HYSTERESIS 20.0000E-3 indicating the hysteresis is set to 20 ms.

BUS:B<x>:CAN:BITRate

This command sets or queries the CAN bitrate. The bus is specified by x.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:CAN:BITRate {RATE10K|RATE100K|RATE1M|RATE125K
| RATE153K|RATE20K|RATE25K|RATE250K
| RATE31K|RATE33K|RATE37K|RATE400K |
RATE50K|RATE500K|RATE62K|RATE68K |
RATE800K|RATE83K|RATE92K|CUSTom}
BUS:B<x>:CAN:BITRate?
```

Arguments Arguments specify the bitrate.

Examples `BUS:B1:CAN:BITRate Rate1M` sets the CAN bitrate to 1 Mb.

`BUS:B1:CAN:BITRate?` might return `:BUS:B1:CAN:BITRATE RATE500K`, indicating the CAN bitrate is set to 500 k.

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

This command sets or queries CAN custom bitrate value. The bus is specified by x.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:CAN:BITRate:VALue <nr3>
BUS:B<x>:CAN:BITRate:VALue?
```

Arguments `<nr3>` specifies the CAN custom bitrate value.

Examples `BUS:B1:CAN:BITRate:VALue 400000` sets the custom bitrate value to 400 k.

`BUS:B1:CAN:BITRate:VALue?` might return `:BUS:B1:CAN:BITRATE:VALUE 500.0000E+3`, indicating that the custom bitrate value is set to 500,000.

BUS:B<x>:CAN:PRObe

This command sets or queries CAN probe type. The bus is specified by x.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:CAN:PRObe {DIFFerential|CANH|CANL|RX|TX}
BUS:B<x>:CAN:PRObe?
```

Arguments DIFFerential specifies a differential probe.

CANH specifies a CANH probe type.

CANL specifies a CANL probe type.

RX specifies an RX probe type.

TX specifies a TX probe type.

Examples `BUS:B1:CAN:PRObe DIFF` sets the probe type to differential.

`BUS:B1:CAN:PRObe?` might return `:BUS:B1:CAN:PROBE CANH`, indicating that the CANH is the probe type.

BUS:B<x>:CAN:SOUrce

This command sets or queries the CAN source channel. The bus is specified by x.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:CAN:SOUrce {CH1|CH2|CH3|CH4 |D0|D1|D10|D11
|D12|D13|D14|D15 |D2|D3|D4|D5 |D6|D7|D8|D9
|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:CAN:SOUrce?
```

Arguments Arguments specify a channel.

Examples `BUS:B1:CAN:SOUrce MATH4` sets the source channel to MATH4.

`BUS:B1:CAN:SOUrce?` might return `:BUS:B1:CAN:SOURCE CH1`, indicating that the CAN source channel is CH1.

BUS:B<x>:DISplay:DECOde:FILe

NOTE. *Not available on some models.*

This command sets or queries the name of a TSF-formatted text file used to construct a symbolic decode table from for the specified bus.

The decode table associates symbolic names with the numeric values they represent. When a Bus is set to perform symbolic decode, numeric values in its "busform" trace are replaced by the first matching symbol (if any) from the table. Each Bus can use its own specific lookup table. For example, an I2C-type Bus might use a table that associates the symbol "ROM" with the number 0b10100111, even though another the decode table for another bus associates the symbol "MAX" with that same number. The lookup table format is the same for all the bus types.

The decode files for all the buses are available in the default location C:\Users\Public\Tektronix\TekScope\busDecodeTables\.

Group Bus

Syntax

```
BUS:B<x>:DISplay:DECOde:FILe {decodeFileName}
BUS:B<x>:DISplay:DECOde:FILe?
```

Arguments `decodeFileName` specifies the name of a TSF-formatted text file.

Examples

`BUS:B1:DISPLAY:DECODE:FILE "C:\Users\Public\Tektronix\TekScope\busDecodeTables\BusDecode.tsf"` uses the TSF-formatted text file BusDecode.tsf.

`BUS:B1:DISPLAY:DECODE:FILE?` might return `:BUS:B1:DISPLAY:DECODE:FILE "C:\Users\Public\Tektronix\TekScope\busDecodeTables\BusDecode.tsf"`, indicating that the TSF-formatted text file used is BusDecode.tsf.

BUS:B<x>:DISplay:DECOde:STAte

NOTE. *Not available on some models.*

This command sets or queries whether the specified bus is enabled to display symbolic decode of its busform values.

Group Bus

Syntax
```
BUS:B<x>:DISplay:DECOde:STAte {OFF|ON|RELoad}
BUS:B<x>:DISplay:DECOde:STAte?
```

Related Commands

Arguments `OFF` = the specified bus does not display symbolic decode of its busform values.

`ON` = the specified bus displays symbolic decode of its busform values.

`RELoad` reparses to whatever symbolic decode file name it is currently assigned. For example, this can be useful if the contents of that file are changed after having assigned it to a Bus.

Examples `BUS:B1:DISPLAY:DECODE:STATE ON` enables the bus B1 to display symbolic decode of its busform values.

`BUS:B1:DISPLAY:DECODE:STATE?` might return `:BUS:B1:DISPLAY:DECODE:STATE OFF`, indicating that the bus B1 does not display symbolic decode of its busform values.

BUS:B<x>:ETHERnet:PRObe

This command specifies the Ethernet probe type: differential or single-ended. The default is `DIFFerential`. `B<x>` specifies the bus number, which can range from 1 to 16.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Bus

Syntax
```
BUS:B<x>:ETHERnet:PRObe {DIFFerential|SINGleended}
BUS:B<x>:ETHERnet:PRObe?
```

Related Commands BUS:B<x>:ETHERnet:SOUrce

BUS:B<x>:ETHERnet:SOUrce:DMINus

BUS:B<x>:ETHERnet:SOUrce:DPLUs

Arguments

`DIFFerential`

`SINGleended`

Examples

`BUS:ETHERNET:PROBE DIFFERENTIAL` specifies that a differential probe is connected to the input channels specified by BUS:B<x>:ETHERnet:SOUrce:DPLUs and BUS:B<x>:ETHERnet:SOUrce:DMINus.

`BUS:ETHERNET:PROBE?` might return `SINGLEENDED`, indicating the probe type has been set to single-ended.

BUS:B<x>:ETHERnet:SOUrce

This command specifies the Ethernet data source for differential input. The supported source waveforms are channels 1–4 and math waveforms 1–4. The default is channel 1. `B<X>` specifies the bus number, which can range from 1 to 16.

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Bus

Syntax

`BUS:B<x>:ETHERnet:SOUrce {CH1|CH2|CH3|CH4|MATH1|MATH2|MATH3|MATH4}`

`BUS:B<x>:ETHERnet:SOUrce?`

Related Commands

BUS:B<x>:ETHERnet:PRObe

Arguments

`CH1–4` specifies to use one of the channels 1–4 as the Ethernet data source for differential input.

`MATH1–4` specifies to use a math waveform as the source for Ethernet data differential input

Examples

`BUS:B1:ETHERNET:SOURCE CH4` specifies to use the channel 4 waveform as the source for Ethernet data.

`BUS:B1:ETHERNET:SOURCE?` might return `CH2`, indicating that channel 2 is the currently specified source for Ethernet data.

BUS:B<x>:ETHERnet:SOUrce:DMINus

This command specifies the Ethernet data source for D- input for differential probing. The default is Channel 2. B<x> specifies the bus number, which can range from 1 to 16.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Bus

Syntax

```
BUS:B<x>:ETHERnet:SOUrce:DMINus {CH1|CH2|CH3|CH4|
D0|D1|D2|D3 |D4| D5|D6|D7|D8| D9|D10|D11|D12| D13|D14|D15|
MATH1|MATH2|MATH3|MATH4}
```

```
BUS:B<x>:ETHERnet:SOUrce:DMINus?
```

Related Commands BUS:B<x>:ETHERnet:PRObe

BUS:B<x>:ETHERnet:SOUrce:DPLUs

Arguments CH1–4 specifies to use one of the analog channels as the Ethernet data source for the D- input.

D0–D15 specifies to use one of the digital channels D0 – D15 as the Ethernet data source for the D- input.

MATH1–4 specifies to use one of the math waveforms as the Ethernet data source for the D- input.

Examples BUS:ETHERNET:SOURCE:DMINUS D5 specifies digital channel 5 as the Ethernet data source for the D- input.

BUS:ETHERNET:SOURCE:DMINUS? might return CH3, indicating that channel 3 is the currently specified Ethernet data source for the D- input.

BUS:B<x>:ETHERnet:SOUrce:DPLUs

This command specifies the Ethernet data source for the D+ input for differential probing. The default is channel 1. B<x> specifies the bus number, which can range from 1 to 16.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Bus

Syntax
```
BUS:B<x>:ETHERnet:SOUrce:DPLUs {CH1|CH2|CH3|CH4|
D0|D1|D2|D3| D4| D5|D6|D7|D8| D9|D10|D11|D12| D13|D14|D15|
MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:ETHERnet:SOUrce:DPLUs?
```

Related Commands BUS:B<x>:ETHERnet:PRObe

BUS:B<x>:ETHERnet:SOUrce:DMINus

Arguments CH1–4 specifies to use one of the analog channels 1 – 4 as the Ethernet data source for the D+ input.

D0–D15 specifies to use one of the digital channels D0 – D15 as the Ethernet data source for the D+ input.

MATH1–4 specifies to use one of the math waveforms as the Ethernet data source for the D+ input.

Examples BUS:ETHERNET:SOURCE:DPLUS D5 specifies D5 as the Ethernet data source for the D+ input.

BUS:ETHERNET:SOURCE:DPLUS? might return CH3, indicating that channel 3 is the currently specified Ethernet data source for the D+ input.

BUS:B<x>:ETHERnet:TYPe

This command specifies the Ethernet standard type: 10Base-T or 100Base-T. The default is ENET100BASETX. B<x> specifies the bus number, which can range from 1 to 16.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Bus

Syntax
```
BUS:B<x>:ETHERnet:TYPe {ENET10BASET|ENET100BASETX}
BUS:B<x>:ETHERnet:TYPe?
```

Related Commands BUS:B<x>:ETHERnet:PRObe

Arguments ENET10BASET specifies the Ethernet type as 10Base-T.

ENET100BASETX specifies the Ethernet type as 100Base-T.

Examples BUS:B1:ETHERNET:TYPE ENET100BASETX specifies the Ethernet type as 100Base-T.

BUS:B1:ETHERNET:TYPE? might return ENET10BASET, indicating that 10Base-T is the currently specified Ethernet type.

BUS:B<x>:FLEXRAY:BITRate

This command sets or queries the FLEXRAY bus bit rate. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:FLEXRAY:BITRate {RATE10M|RATE5M|RATE2M|CUSTom}
BUS:B<x>:FLEXRAY:BITRate?
```

Arguments Arguments specify the bit rate.

Examples BUS:B1:FLEXRAY:BITRate RATE2M sets the bit rate to 2 Mb.

BUS:B1:FLEXRAY:BITRate? might return :BUS:B1:FLEXRAY:BITRATE RATE10M, indicating the bit rate is 10 Mb.

BUS:B<x>:FLEXRAY:BITRate:VALue

This command sets or queries FLEXRAY custom bit rate. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:FLEXRAY:BITRate:VALue <nr3>
BUS:B<x>:FLEXRAY:BITRate:VALue?
```

Arguments <nr3> specifies the custom bit rate value.

Examples `BUS:B1:FLEXRAY:BITRate:VALue 2.0E+6` sets the custom bit rate to 2 Mb.

`BUS:B1:FLEXRAY:BITRate:VALue?` might return `:BUS:B1:FLEXRAY:BITRATE:VALUE 10.0000E+6`, indicating the custom bit rate is set to 10 Mb.

BUS:B<x>:FLEXRAY:CHANnel

This command sets or queries the FLEXRAY bus input channel. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:FLEXRAY:CHANnel {A|B}
BUS:B<x>:FLEXRAY:CHANnel?
```

Arguments `A` specifies the A channel.

`B` specifies the B channel.

Examples `BUS:B1:FLEXRAY:CHANnel B` sets the FLEXRAY channel to B.

`BUS:B1:FLEXRAY:CHANnel?` might return `:BUS:B1:FLEXRAY:CHANNEL A`, indicating the channel is set to A.

BUS:B<x>:FLEXRAY{:PROBe|:SIGnal}

This command sets or queries the FLEXRAY probe. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:FLEXRAY{:PROBe|:SIGnal} {BDIFFBP|BM|TXRX}
BUS:B<x>:FLEXRAY{:PROBe|:SIGnal}?
```

Arguments Arguments are the FLEXRAY probe or signal types.

Examples `BUS:B1:FLEXRAY:PROBe BM` sets the FLEXRAY probe type to BM.

`BUS:B1:FLEXRAY:PROBe?` might return `:BUS:B1:FLEXRAY:PROBE BDIFFBP`, indicating the FLEXRAY probe is set to BDIFFBP.

BUS:B<x>:FLEXRAY:SOUrce

This command sets or queries the FLEXRAY bus signal source. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:FLEXRAY:SOUrce {CH1|CH2|CH3|CH4 |D0|D1|D10|D11
|D12|D13|D14|D15 |D2|D3|D4|D5 |D6|D7|D8|D9
|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:FLEXRAY:SOUrce?
```

Arguments Arguments are the available FLEXRAY sources.

Examples `BUS:B1:FLEXRAY:SOUrce Math4` sets the FLEXRAY source to Math4.

`BUS:B1:FLEXRAY:SOUrce?` might return `:BUS:B1:FLEXRAY:SOURCE CH1`, indicating the source is channel 1.

BUS:B<x>:I2C:CLOCk:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the I2C clock (SCLK) source for the specified bus.

Group Bus

Syntax
```
BUS:B<x>:I2C:CLOCk:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4|}
```

BUS:B<x>:I2C:CLOCk:SOUrce?

Related Commands BUS:B<x>:I2C:DATa:SOUrce, BUS:B<x>:I2C:RWINADDR

Arguments CH1–CH4 specifies an analog channel to use as the I2C SCLK source.

D0–D15 specifies a digital channel to use as the I2C SCLK source.

MATH1–MATH4 specifies a math channel to use as the I2C SCLK source.

Examples BUS:B1:I2C:CLOCK:SOURCE CH1 sets the I2C SCLK source to CH1.

BUS:B1:I2C:CLOCK:SOURCE? might return :BUS:B1:I2C:CLOCK:SOURCE D4, indicating that the I2C SCLK source is set to D4.

BUS:B<x>:I2C:DATa:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the I2C data (SDA) source for the specified I2C bus.

Group Bus

Syntax
```
BUS:B<x>:I2C:DATa:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:I2C:DATa:SOUrce?
```

Related Commands BUS:B<x>:I2C:CLOCk:SOUrce, BUS:B<x>:I2C:RWINADDR

Arguments CH1–CH4 specifies an analog channel to use as the I2C SDATA source.

D0–D15 specifies a digital channel to use as the I2C SDATA source.

MATH1–MATH4 specifies a math channel to use as the I2C SDATA source.

Examples BUS:B1:I2C:DATA:SOURCE D5 sets the I2C SDATA source to D5.

BUS:B1:I2C:DATA:SOURCE? might return :BUS:B1:I2C:DATA:SOURCE MATH1, indicating that the I2C SDATA source is set to MATH1.

BUS:B<x>:I2C:RWINADDR

NOTE. *Not available on some models.*

This command sets or queries the manner in which seven-bit I2C slave addresses are represented in the "busform" display of the specified bus.

Group Bus

Syntax

```
BUS:B<x>:I2C:RWINADDR {No|Yes}
BUS:B<x>:I2C:RWINADDR?
```

Related Commands BUS:B<x>:I2C:CLOCk:SOUrce, BUS:B<x>:I2C:DATa:SOUrce

Arguments

`NO` displays seven-bit slave addresses as integers in the range of 0 to 127, with the state of the R/W* bit from the LSB of the slave address byte. For example, the slave address byte of 0b10100101 is displayed as the value 0x52 R.

`Yes` displays the entire slave address byte as a number, with the R/W* signal as its LSB (bit 0) and the slave address in bits 7..0. For example, the slave address byte of 0b10100101 is displayed as the value 0xA5 R.

Examples

`BUS:B1:I2C:RWINADDR NO` displays seven-bit slave addresses as integers in the range of 0 to 127.

`BUS:B1:I2C:RWINADDR?` might return `:BUS:B1:I2C:RWINADDR YES`, indicating that the entire slave address byte is displayed as a number, with the R/W* signal as its LSB (bit 0) and the slave address in bits 7..0.

BUS:B<x>:LABel

NOTE. *Not available on some models.*

This command sets or queries the waveform label for the specified bus. The bus name string accepts only eight characters and truncates when more than eight characters.

Group Bus

Syntax

```
BUS:B<x>:LABel <string>
BUS:B<x>:LABel?
```

Related Commands BUS:B<x>:POSition, BUS:B<x>:TYPe

Arguments <string> is an alphanumeric string of text enclosed in quotes. The text string is limited to 30 characters. It contains the text label information for bus.

Examples BUS:B1:LABEL "TEST" sets the waveform label for the bus B1 to Test.

BUS:B1:LABEL? might return :BUS:B1:LABEL "BUS 1", indicating that the waveform label for the bus B1 is set to "Bus 1".

BUS:B<x>:LIN:BITRate

This command sets or queries the LIN bus bit rate. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:LIN:BITRate
{RATE10K|RATE1K|RATE19K|RATE2K|RATE4K|RATE9K|CUSTom}
BUS:B<x>:LIN:BITRate?
```

Arguments Arguments are the available bit rates.

Examples BUS:B1:LIN:BITRate Rate4k sets the bit rate to 4 kb.

BUS:B1:LIN:BITRate? might return :BUS:B1:LIN:BITRATE RATE19K, indicating that the bit rate is set to 19 kb.

BUS:B<x>:LIN:BITRate:VALue

This command sets or queries the LIN bus custom bitrate value. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:LIN:BITRate:VALue <nr3>
BUS:B<x>:LIN:BITRate:VALue?
```

Arguments <nr3> specifies the custom bit rate.

Examples `BUS:B1:LIN:BITRate:VALue 19200` sets the custom bit rate value to 19.2 k.

`BUS:B1:LIN:BITRate:VALue?` might return `:BUS:B1:LIN:BITRATE:VALUE 19.2000E+3`, indicating the custom bit rate value is 19.2 k.

BUS:B<x>:LIN:IDFORmat

This command sets or queries LIN bus id display mode. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:LIN:IDFORmat {NOPARity|PARity}
BUS:B<x>:LIN:IDFORmat?
```

Arguments `NOPARity` specifies an id format that includes parity.

`PARity` specifies an id format that separates parity.

Examples `BUS:B1:LIN:IDFORmat PARITY` set the id format to parity.

`BUS:B1:LIN:IDFORmat?` might return `:BUS:B1:LIN:IDFORMAT NOPARITY`, indicating the id format is set to no parity.

BUS:B<x>:LIN:POLarity

This command sets or queries the LIN bus polarity. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:LIN:POLarity {INVerted|NORMal}
BUS:B<x>:LIN:POLarity?
```

Arguments INVerted specifies inverted polarity.

NORMal specifies normal polarity.

Examples `BUS:B1:LIN:POLarity Inverted` sets the polarity to inverted.

`BUS:B1:LIN:POLarity?` might return `:BUS:B1:LIN:POLARITY NORMAL`, indicating the bus polarity is set to normal.

BUS:B<x>:LIN:SOUrce

This command sets or queries sets the LIN bus source. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:LIN:SOUrce {CH1|CH2|CH3|CH4 |D0|D1|D10|D11
|D12|D13|D14|D15 |D2|D3|D4|D5 |D6|D7|D8|D9
|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:LIN:SOUrce?
```

Arguments Arguments specify the available sources.

Examples `BUS:B1:LIN:SOUrce MATH4` sets the source to MATH 4

`BUS:B1:LIN:SOUrce?` might return `:BUS:B1:LIN:SOURCE CH1`, indicating the source is set to channel 1.

BUS:B<x>:LIN:STANDard

This command sets or queries the LIN bus standard. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

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

Arguments `MIXed` specifies V1X and V2X.

`V1X` specifies the V1X standard.

`V2X` specifies the V2X standard.

Examples `BUS:B1:LIN:STANDard V1X` sets the standard to V1X.

`BUS:B1:LIN:STANDard?` might return `:BUS:B1:LIN:STANDARD V2X`, indicating the standard is set to V2X.

BUS:B<x>:MIL1553B:POLarity

This command sets the MIL-STD-1553 bus polarity to normal or inverted. B<x> specifies the bus number, which can range from 1 to 16.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Bus

Syntax
```
BUS:B<x>:MIL1553B:POLarity {NORMal|INVerted}
BUS:B<x>:MIL1553B:POLarity?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

BUS:B<x>:MIL1553B:SOUrce

Arguments `NORMal` – A high-low transition sets the MIL-STD-1553 bus polarity to positive.

`INVerted` – A high-low transition sets the MIL-STD-1553 bus polarity to negative.

Examples `BUS:B1:MIL1553B:POLARITY INVERTED` sets the MIL-STD-1553 bus polarity to negative.

`BUS:B1:MIL1553B:POLARITY?` might return NORMAL.

BUS:B<x>:MIL1553B:RESPonsetime:MAXimum

This command specifies the maximum response time to a valid command issued. B<x> specifies the bus number, which can range from 1 to 16.

NOTE. *The MIL-STD-1553 bus specification requires devices to respond to a valid command within 4 to 12 microseconds.*

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Bus

Syntax

```
BUS:B<x>:MIL1553B:RESPonsetime:MAXimum <NR3>
BUS:B<x>:MIL1553B:RESPonsetime:MAXimum?
```

Related Commands BUS:B<x>:MIL1553B:RESPonsetime:MINimum

Arguments <NR3> is a floating point number that specifies the maximum response time, in seconds.

Examples BUS:B1:MIL1553B:RESPONSETIME:MAXIMUM 12.0E-6 specifies the maximum response time to a valid command received to be 12 microseconds.

BUS:B1:MIL1553B:RESPONSETIME:MAXIMUM? might return 12.0000E-6.

BUS:B<x>:MIL1553B:RESPonsetime:MINimum

This command specifies the minimum response time to a valid command issued. B<x> specifies the bus number, which can range from 1 to 16.

NOTE. *The MIL-STD-1553 bus specification requires devices to respond to a valid command within 4 to 12 microseconds.*

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Bus

Syntax

```
BUS:B<x>:MIL1553B:RESPonsetime:MINimum <NR3>
BUS:B<x>:MIL1553B:RESPonsetime:MINimum?
```

Related Commands BUS:B<x>:MIL1553B:RESPonsetime:MAXimum

Arguments <NR3> is a floating point number that specifies the minimum response time in seconds.

Examples BUS:B1:MIL1553B:RESPONSETIME:MINIMUM 4.0E-6 specifies the minimum response time to a valid command received to 4 microseconds.

BUS:B1:MIL1553B:RESPONSETIME:MINIMUM? might return 4.0000E-6.

BUS:B<x>:MIL1553B:SOUrce

This command sets or queries sets the MIL-STD-1553 bus source. The supported source waveforms are channels 1-4 and math waveforms 1-4. The default is channel 1. B<x> specifies the bus number, which can range from 1 to 16.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Bus

Syntax

```
BUS:B<x>:MIL1553B:SOUrce
{CH1|CH2|CH3|CH4|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:MIL1553B:SOUrce?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:CONDition

Arguments CH1–4 specifies to use one of the analog channels as the MIL-STD-1553 bus source for differential input.

MATH1–4 specifies to use the math waveform as the MIL-STD-1553 bus source for differential input

Examples BUS:B1:MIL1553B:SOURCE CH4 specifies to use analog channel 4 waveform as the source for MIL-STD-1553 bus input.

BUS:B1:MIL1553B:SOURCE? might return CH2, indicating that channel 2 is the currently specified source for MIL-STD-1553 bus input.

BUS:B<x>:MIPICSITWo:CLOCk:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the MIPI CSI2 clock source for the specified bus.

Group Bus

Syntax

```
BUS:B<x>:MIPICSITWo:CLOCk:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:MIPICSITWo:CLOCk:SOUrce?
```

Related Commands BUS:B<x>:MIPICSITWo:CLOCk:TYPe

Arguments

`CH1–CH4` specifies an analog channel to use as the MIPI CSI2 clock source.

`D0–D15` specifies a digital channel to use as the MIPI CSI2 clock source.

`MATH1–MATH4` specifies a math channel to use as the MIPI CSI2 clock source.

Examples

`BUS:B1:MIPICSITWO:CLOCK:SOURCE CH1` sets the MIPI CSI2 clock source for bus B1 to CH1.

`BUS:B1:MIPICSITWO:CLOCK:SOURCE?` might return `:BUS:B1:MIPICSITWO:CLOCK:SOURCE MATH1`, indicating that the MIPI CSI2 clock source for bus B1 is set to MATH1.

BUS:B<x>:MIPICSITWo:CLOCk:TYPe

NOTE. *Not available on some models.*

This command sets or queries the MIPI CSI2 clock source type for the specified bus.

Group Bus

Syntax

```
BUS:B<x>:MIPICSITWo:CLOCk:TYPe {ANALog|DIGItal}
BUS:B<x>:MIPICSITWo:CLOCk:TYPe?
```

Related Commands BUS:B<x>:MIPICSITWo:CLOCk:SOUrce

Arguments `ANALog` sets the clock source type to support analog input for the specified bus.

`DIGItal` sets the clock source type to support digital input for the specified bus.

Examples `BUS:B1:MIPICSITWO:CLOCK:TYPE ANALOG` sets the clock source type for bus B1 to analog.

`BUS:B1:MIPICSITWO:CLOCK:TYPE?` might return `:BUS:B1:MIPICSITWO:CLOCK:TYPE DIGITAL`, indicating that the clock source type for bus B1 is digital.

BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DIFFerential

NOTE. *Not available on some models.*

This command sets or queries the differential source for the specified lane of the specified MIPI CSI2 bus.

Group Bus

Syntax
```
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DIFFerential
{D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DIFFerential?
```

Related Commands BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DMINUS, BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DPLUS, BUS:B<x>:MIPICSITWo:LANE<x>:TYPe

Arguments `D0-D15` specifies a digital input signal to use as the differential source.

Examples `BUS:B1:MIPICSITWO:LANE1:SOURCE:DIFFERENTIAL D2` sets the differential source for lane 1 of MIPI CSI2 bus B1 to D2.

`BUS:B1:MIPICSTWO:LANE2:SOURCE:DIFFERENTIAL?` might return `:BUS:B1:MIPICSITWO:LANE2:SOURCE:DIFFERENTIAL D3`, indicating that the differential source for lane 2 of MIPI CSI2 bus B1 is D3.

BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DMINUS

NOTE. *Not available on some models.*

This command sets or queries the D Minus source for the specified lane of the specified MIPI CSI2 bus.

Group Bus

Syntax

```
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DMINUS
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DMINUS?
```

Related Commands BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DIFFerential, BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DPLUS, BUS:B<x>:MIPICSITWo:LANE<x>:TYPe

Arguments

`CH1–CH4` specifies an analog channel to use as the D Minus source for the specified lane.

`D0–D15` specifies a digital channel to use as the D Minus source for the specified lane.

`MATH1–MATH4` specifies a math channel to use as the D Minus source for the specified lane.

Examples

`BUS:B1:MIPICSITWO:LANE1:SOURCE:DMINUS CH4` sets the D Minus source to CH4 for lane 1 of bus B1.

`BUS:B1:MIPICSITWO:LANE2:SOURCE:DMINUS?` might return `:BUS:B1:MIPICSITWO:LANE2:SOURCE:DMINUS CH3`, indicating that the D Minus source for lane 2 of bus B1 is D3.

BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DPLUS

NOTE. *Not available on some models.*

This command sets or queries the D Plus source for the specified lane of the specified MIPI CSI2 bus.

Group Bus

Syntax

```
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DPLUS
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DPLUS?
```

Related Commands BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DIFFerential, BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DMINUS, BUS:B<x>:MIPICSITWo:LANE<x>:TYPe

Arguments

`CH1–CH4` specifies an analog channel to use as the D Plus source for the specified lane.

`D0–D15` specifies a digital channel to use as the D Plus source for the specified lane.

`MATH1–MATH4` specifies a math channel to use as the D Plus source for the specified lane.

Examples

`BUS:B1:MIPICSITWO:LANE1:SOURCE:DPLUS CH3` sets the D Plus source for Lane 1 to CH3 for bus B1.

`BUS:B1:MIPICSITWO:LANE1:SOURCE:DPLUS?` might return `:BUS:B1:MIPICSITWO:LANE1:SOURCE:DPLUS CH4`, indicating that the D Plus source for Lane 1 is CH4 for the bus B1.

BUS:B<x>:MIPICSITWo:LANE<x>:TYPe

NOTE. *Not available on some models.*

This command sets or queries the lane source type for the specified MIPI CSI2 bus.

Group Bus

Syntax

```
BUS:B<x>:MIPICSITWo:LANE<x>:TYPe {ANALog|DIGItal}
BUS:B<x>:MIPICSITWo:LANE<x>:TYPe?
```

Related Commands BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DIFFerential, BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DMINUS, BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:DPLUS,

Arguments

ANALog sets the source type for the specified lane to support analog input.

DIGItal sets the source type for the specified lane to support digital input.

Examples

BUS:B1:MIPICSITWO:LANE1:TYPE ANALOG sets the lane 2 source type for bus B1 to analog.

BUS:B2:MIPICSITWO:LANE2:TYPE? might return :BUS:B2:MIPICSITWO:LANE2 DIGITAL, indicating that the lane 2 source type for bus B2 is set to digital.

BUS:B<x>:MIPIDSIOne:CLOCk:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the clock source for the specified MIPI DSI1 bus.

Group

Bus

Syntax

```
BUS:B<x>:MIPIDSIOne:CLOCk:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:MIPIDSIOne:CLOCk:SOUrce?
```

Related Commands

BUS:B<x>:MIPIDSIOne:CLOCk:TYPe

Arguments

CH1–CH4 specifies an analog channel to use as the clock source.

MATH1–MATH4 specifies a math channel to use as the clock source.

D0–D15 specifies a digital channel to use as the clock source.

Examples

BUS:B1:MIPIDSIONE:CLOCK:SOURCE CH2 sets the clock source for bus B1 to CH2.

BUS:B1:MIPIDSIONE:CLOCK:SOURCE? might return :BUS:B1:MIPIDSIONE:CLOCK:SOURCE CH4, indicating that the clock source for MIPI CSI1 bus B1 is set to CH4.

BUS:B<x>:MIPIDSIOne:CLOCk:TYPe

NOTE. *Not available on some models.*

This command sets or queries the clock source type for the specified MIPI DSI1 bus.

Group Bus

Syntax

```
BUS:B<x>:MIPIDSIOne:CLOCk:TYPe {ANALog|DIGItal}
BUS:B<x>:MIPIDSIOne:CLOCk:TYPe?
```

Related Commands BUS:B<x>:MIPIDSIOne:CLOCk:SOUrce

Arguments

`ANALog` sets the clock source type to support analog input.

`DIGItal` sets the clock source type to support digital input.

Examples

`BUS:B1:MIPIDSIONE:CLOCK:TYPE DIGITAL` sets the clock source type for MIPI DSI1 bus B1 to digital.

`BUS:B1:MIPIDSIONE:CLOCK:TYPE?` might return `:BUS:B1:MIPIDSIONE:CLOCK:TYPE ANALOG`, indicating that the clock source type for MIPI DSI1 bus B1 is analog.

BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DIFFerential

NOTE. *Not available on some models.*

This command sets or queries the differential source for the specified lane of the specified MIPI DSI1 bus.

Group Bus

Syntax

```
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DIFFerential
{D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15}
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DIFFerential?
```

Related Commands BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DMINUS, BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DPLUS, BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe

Arguments D0–D15 specifies a digital input signal to use as the differential source for the specified lane.

Examples BUS:B1:MIPIDSIONE:LANE1:SOURCE:DIFFERENTIAL D5 sets the differential source for lane 1 of MIPI DSI1 bus B1 to D5.

BUS:B1:MIPIDSIONE:LANE1:SOURCE:DIFFERENTIAL? might return :BUS:B1:MIPIDSIONE:LANE1:SOURCE:DIFFERENTIAL D4, indicating that the differential source for lane 1 of MIPI DSI1 bus B1 is D4.

BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DMINUS

NOTE. *Not available on some models.*

This command sets or queries the D Minus source for the specified lane of the specified MIPI DSI1 bus.

Group Bus

Syntax

```
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DMINUS
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DMINUS?
```

Related Commands BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DIFFerential, BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DPLUS, BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe

Arguments CH1–CH4 specifies an analog channel to use as the D Minus source for the specified lane.

D0–D15 specifies a digital channel to use as the D Minus source for the specified lane.

MATH1–MATH4 specifies a math channel to use as the D Minus source for the specified lane.

Examples

`BUS:B1:MIPIDSIONE:LANE1:SOURCE:DMINUS MATH4` sets the D Minus source for lane 1 of MIPI DSI1 bus B1 to MATH4.

`BUS:B1:MIPIDSIONE:LANE2:SOURCE:DMINUS?` might return `:BUS:B1:MIPIDSIONE:LANE2:SOURCE:DMINUS CH1`, indicating that the D Minus source for lane 2 of MIPI DSI1 bus B1 is CH1.

BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DPLUS

NOTE. *Not available on some models.*

This command sets or queries the D Plus source for the specified lane of the specified MIPI DSI1 bus.

Group

Bus

Syntax

```
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DPLUS
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DPLUS?
```

Related Commands

BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DIFFerential, BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DMINUS, BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe

Arguments

`CH1–CH4` specifies an analog channel to use as the D Plus source for the specified lane.

`D0–D15` specifies a digital channel to use as the D Plus source for the specified lane.

`MATH1–MATH4` specifies a math channel to use as the D Plus source for the specified lane.

Examples

`BUS:B1:MIPIDSIONE:LANE1:SOURCE:DPLUS D2` sets the D Plus source for lane 1 of MIPI DSI1 bus B1 to D2.

`BUS:B1:MIPIDSIONE:LANE2:SOURCE:DPLUS?` might return `:BUS:B1:MIPIDSIONE:LANE2:SOURCE:DPLUS CH4`, indicating that the D Plus source for lane 2 of MIPI DSI1 bus B1 is CH4.

BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe

NOTE. *Not available on some models.*

This command sets or queries the lane source type for the specified MIPI DSI1 bus.

Group Bus

Syntax

```
BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe {ANALog|DIGItal}
BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe?
```

Related Commands BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DIFFerential, BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DMINUS, BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:DPLUS

Arguments `ANALog` sets the source type for the specified lane to support analog input.

`DIGItal` sets the source type for the specified lane to support digital input.

Examples `BUS:B1:MIPIDSIONE:LANE1:TYPE ANALOG` sets the lane 1 type for MIPI DSI1 bus B1 to analog.

`BUS:B1:MIPIDSIONE:LANE1:TYPE?` might return `:BUS:B1:MIPIDSIONE:LANE1:TYPE DIGITAL`, indicating that the lane 1 type of MIPI DSI1 bus B1 is digital.

BUS:B<x>:PARallel:CLOCk:EDGE

NOTE. *Not available on some models.*

This command sets or queries which edge of a clocked parallel busses' signal establishes when new bus values are sampled.

Group Bus

Syntax

```
BUS:B<x>:PARallel:CLOCk:EDGE {FALL|RISe|EITHer}
BUS:B<x>:PARallel:CLOCk:EDGE?
```

Related Commands BUS:B<x>:PARallel:CLOCk:SOUrce, BUS:B<x>:PARallel:ISCLOCKED, BUS:B<x>:PARallel:SOURCES

Arguments `FALL` decodes on the falling edge of the clocked parallel bus signal.

`RISe` decodes on the rising edge of the clocked parallel bus signal.

`EITHer` decodes on the rising or falling edge of the clocked parallel bus signal.

Examples `BUS:B1:PARALLEL:CLOCK:EDGE FALL` sets the decoding to happen on the falling edge of its clocked parallel bus B1.

`BUS:B1:PARALLEL:CLOCK:EDGE?` might return `:BUS:B1:PARALLEL:CLOCK:EDGE RISE`, indicating that when bus B1 operates in Clocked Parallel mode, it samples new data values on the rising edge of its clock source signal.

BUS:B<x>:PARallel:CLOCk:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the Parallel clock source for the specified bus.

Group Bus

Syntax
```
BUS:B<x>:PARallel:CLOCk:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:PARallel:CLOCk:SOUrce?
```

Related Commands BUS:B<x>:PARallel:CLOCk:EDGE, BUS:B<x>:PARallel:ISCLOCKED, BUS:B<x>:PARallel:SOURCES

Arguments `CH1–CH4` specifies an analog channel to use as the Parallel clock source.

`D0–D15` specifies a digital channel to use as the Parallel clock source.

`MATH1–MATH4` specifies the math channel to use as the Parallel clock source.

Examples `BUS:B1:PARALLEL:CLOCK:SOURCE D6` sets the Parallel clock source for the bus B1 to D6.

BUS:B1:PARALLEL:CLOCK:SOURCE? might return :BUS:B1:PARALLEL:CLOCK:SOURCE D11, indicating that the Parallel clock source for the bus B1 is set to D11.

BUS:B<x>:PARallel:ISCLOCKED

NOTE. *Not available on some models.*

This command sets or queries the Parallel bus behavior to either Clocked or not Clocked for the specified bus.

Group Bus

Syntax

```
BUS:B<x>:PARallel:ISCLOCKED {YES|NO}
BUS:B<x>:PARallel:ISCLOCKED?
```

Related Commands BUS:B<x>:PARallel:CLOCk:EDGE, BUS:B<x>:PARallel:CLOCk:SOUrce, BUS:B<x>:PARallel:SOURCES

Arguments YES specifies the Parallel mode of the specified bus to be clocked.

NO specifies the Parallel mode of the specified bus to be unclocked (asynchronous).

Examples BUS:B1:PARALLEL:ISCLOCKED YES sets B1 bus to have Clocked behavior when operating in Parallel mode.

BUS:B1:PARALLEL:ISCLOCKED? might return :BUS:B1:PARALLEL:ISCLOCKED YES, indicating that the bus B1 is set to Clocked Parallel mode.

BUS:B<x>:PARallel:SOURCES

NOTE. *Not available on some models.*

This command sets or queries the members of the Parallel mode of specified bus according to a supplied list of signals.

The first item on the list becomes the MSB signal of the Bus. The second becomes the next-most-significant, and so on, with the last item becoming the LSB of the bus. If no signals are listed, the Bus is emptied of members.

No signal can appear more than once in the list of bus members.

Group Bus

Syntax

```
BUS:B<x>:PARallel:SOURCES
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:PARallel:SOURCES?
```

Related Commands BUS:B<x>:PARallel:CLOCk:EDGE, BUS:B<x>:PARallel:CLOCk:SOUrce, BUS:B<x>:PARallel:ISCLOCKED

Arguments CH1–CH4 specifies an analog channel to use as the Parallel source.

D0–D15 specifies a digital channel to use as the Parallel source.

MATH1–MATH4 specifies the math channel to use as the Parallel source.

Examples BUS:B1:PARALLEL:SOURCES D3, D1, D4, D2 sets the digital signals D3, D1, D4, and D2 as members of the Parallel bus B1.

BUS:B1:PARALLEL:SOURCES? might return :BUS:B1:PARALLEL:SOURCES D3, D5, indicating that the digital signals D3 and D5 are part of the Parallel bus B1.

BUS:B<x>:PCIE:BITRate

This command sets or queries the PCIE bus bit rate. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:PCIE:BITRate
{RATE2500|RATE5000|RATE8000|RATE16000|AUTO|CUSTom}
BUS:B<x>:PCIE:BITRate?
```

Arguments Arguments are the available bit rates.

Examples `BUS:B1:PCIE:BITRate RATE8000` sets the bitrate to 8.0 G.

`BUS:B1:PCIE:BITRate?` might return `:BUS:B1:PCIE:BITRATE RATE2500`, indicating the bit rate is 2.5 G.

BUS:B<x>:PCIE:BITRate:VALue

This command sets or queries the PCIE bus bit rate value. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:PCIE:BITRate:VALue <nr3>
BUS:B<x>:PCIE:BITRate:VALue?
```

Arguments `<nr3>` specifies the bit rate value.

Examples `BUS:B1:PCIE:BITRate:VALue 1.5E+9` sets the bit rate value to 1.5 Gb.

`BUS:B1:PCIE:BITRate:VALue?` might return `:BUS:B1:PCIE:BITRATE:VALUE 2.5000E+9`, indicating the bit rate value is 2.5 Gb.

BUS:B<x>:PCIE:HYSTeresis

This command sets or queries the PCIE bus hysteresis. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:PCIE:HYSTeresis <nr3>
BUS:B<x>:PCIE:HYSTeresis?
```

Arguments `<nr3>` specifies the hysteresis.

Examples `BUS:B1:PCIE:HYSTeresis 0.02` sets the hysteresis to 20 mV.

`BUS:B1:PCIE:HYSTeresis?` might return `:BUS:B1:PCIE:HYSTERESIS 50.0000E-3`, indicating the hysteresis is 50 mV.

BUS:B<x>:PCIE:LANE

This command sets or queries the PCIE bus lane number. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:PCIE:LANE <nr3>
BUS:B<x>:PCIE:LANE?
```

Arguments `<nr3>` specifies the bus lane.

Examples `BUS:B1:PCIE:LANE 1` sets the bus lane to 1.

`BUS:B1:PCIE:LANE?` might return `:BUS:B1:PCIE:LANE 0`, indicating bus lane 0.

BUS:B<x>:PCIE:SOUrce

This command sets or queries the PCIE bus source. The bus is specified by x. The value of x can range from 1 to 16.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:PCIE:SOUrce {CH1|CH2|CH3|CH4 | D0|D1|D10|D11
| D12|D13|D14|D15 | D2|D3|D4|D5 | D6|D7|D8|D9 |
MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:PCIE:SOUrce?
```

Arguments Arguments are the available sources.

Examples

`BUS:B1:PCIE:SOUrce MATH4` sets the PCIE bus source to MATH4.

`BUS:B1:PCIE:SOUrce?` might return `:BUS:B1:PCIE:SOURCE CH1`, indicating the PCIE bus source is channel 1.

BUS:B<x>:POSition

NOTE. *Not available on some models.*

This command sets or queries the position for the waveform specified bus.

Group

Bus

Syntax

```
BUS:B<x>:POSition <NR3>
BUS:B<x>:POSition?
```

Related Commands

BUS:B<x>:LABel, BUS:B<x>:TYPe

Arguments

<NR3> specifies the position.

Examples

`BUS:B1:POSITION 3` sets the position of the busform of the bus B1 to 3 Divisions. (above the zero level).

`BUS:B1:POSITION?` might return `:BUS:B1:POSITION 5.0000`, indicating that the position of the busform of the bus B1 is set to 5 Divisions. (above the zero level).

BUS:B<x>:RS232C:BITRate

NOTE. *Not available on some models.*

This command sets or queries the RS232C bit rate for bus<x>, where x is the bus number.

Group

Bus

Syntax

```
BUS:B<x>:RS232C:BITRate <NR3>
BUS:B<x>:RS232C:BITRate?
```

Arguments <NR3> is the bit rate in bits-per-second. You can enter any positive integer, and the instrument will coerce the value to the closest supported bit rate.

Examples BUS:B1:RS232C:BITRATE 9500 sets the bit rate for the RS232C bus B1 to 9500 bits-per-second.

BUS:B1:RS232C:BITRATE? might return :BUS:B1:RS232C:BITRATE 9.5000E+3, indicating that the bit rate for the RS232C bus B1 is set to 9500 bits-per-second.

BUS:B<x>:RS232C:DATABits

NOTE. *Not available on some models.*

This command sets or queries the number of RS-232C data bits for bus<x>, where x is the bus number.

Group Bus

Syntax
```
BUS:B<x>:RS232C:DATABits <NR3>
BUS:B<x>:RS232C:DATABits?
```

Arguments <NR3> specifies the number of bits in the RS-232C data frame.

Examples BUS:B1:RS232C:DATABITS 8 sets the data bits to 8 for the RS-232C bus B1.

BUS:B1:RS232C:DATABITS? might return :BUS:B1:RS232C:DATABITS 8, indicating that the data bits is set to 8 for the RS-232C bus B1.

BUS:B<x>:RS232C:DELIMiter

NOTE. *Not available on some models.*

This command sets or queries the RS-232C delimiting value for a packet on bus <x>, where x is the bus number.

Group Bus

Syntax

```
BUS:B<x>:RS232C:DELIMiter {NUL1|CR|LF|SPace|XFF}
BUS:B<x>:RS232C:DELIMiter?
```

Arguments

`NULl` specifies NULl (0x00) delimiting value for a packet.

`CR` specifies CR (0x0D) delimiting value for a packet.

`LF` specifies LF (0x0A) delimiting value for a packet.

`XFF` specifies XFF (0xFF) delimiting value for a packet.

`SPace` specifies SPace delimiting value for a packet.

Examples

`BUS:B1:RS232C:DELIMITER LF` sets the delimiter to 0x0A for the RS-232C bus B1.

`BUS:B1:RS232C:DELIMITER?` might return `:BUS:B1:RS232C:DELIMITER NULL`, indicating that the delimiter to 0x00 for the RS-232C bus B1.

BUS:B<x>:RS232C:DISplaymode

NOTE. *Not available on some models.*

This command sets or queries the RS-232 display mode for the bus <x> display and event table, where x is the bus number.

Group

Bus

Syntax

```
BUS:B<x>:RS232C:DISplaymode {FRAme|PACKET}
BUS:B<x>:RS232C:DISplaymode?
```

Arguments

`FRAme` displays each frame as a single entity.

`PACKET` displays a group of frames terminated with a single frame defined by the `BUS:B<x>:RS232C:DELImiter` command or the front panel.

Examples

`BUS:B1:RS232C:DISPLAYMODE FRAME` sets the display mode for the RS-232C bus B1 to Frame.

`BUS:B1:RS232C:DISPLAYMODE?` might return `:BUS:B1:RS232C:DISPLAYMODE PACKET`, indicating that the display mode for the RS-232C bus B1 is set to Packet.

BUS:B<x>:RS232C:PARity

NOTE. *Not available on some models.*

This command sets or queries the RS-232C parity for bus <x>, where x is the bus number.

Group Bus

Syntax

```
BUS:B<x>:RS232C:PARity {NONe|EVEN|ODD}
BUS:B<x>:RS232C:PARity?
```

Arguments

`NONe` specifies no parity.

`EVEN` specifies even parity.

`ODD` specifies odd parity.

Examples

`BUS:B1:RS232C:PARITY ODD` sets the parity for the RS-232C bus B1 to odd.

`BUS:B1:RS232C:PARITY?` might return `:BUS:B1:RS232C:PARITY NONE`, indicating that the display mode for the RS-232C bus B1 is set to none.

BUS:B<x>:RS232C:POLarity

NOTE. *Not available on some models.*

This command sets or queries the RS-232 polarity for bus <x>, where x is the bus number.

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.

Examples `BUS:B1:RS232C:POLARITY inverted` sets the polarity for the RS232C bus B1 to Inverted.

`BUS:B1:RS232C:POLARITY?` might return `:BUS:B1:RS232C:POLARITY NORMAL`, indicating that the polarity for the RS232C bus B1 is set to Normal.

BUS:B<x>:RS232C:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the RS-232 polarity for bus <x>, where x is the bus number.

Group Bus

Syntax

```
BUS:B<x>:RS232C:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:RS232C:SOUrce?
```

Arguments `CH1–CH4` specifies an analog channel to use as the RS232C source.

`D0–D15` specifies a digital channel to use for the RS232C source.

`MATH1–MATH4` specifies a math channel to use for the RS232C source.

Examples `BUS:B1:RS232C:SOURCE D0` sets the source for the RS232C bus B1 to D0.

`BUS:B1:RS232C:SOURCE?` might return `:BUS:B1:RS232C:SOURCE CH1`, indicating that the source for the RS-232 bus B1 is CH1.

BUS:B<x>:S8B10B:BITRate

Sets the bus data for the specified bus to a standard rate in bits per second, or enables you to specify a custom data rate using the value command. The query returns the bus data rate that is set for the specified bus.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:S8B10B:BITRate
{CUSTom|RATE1250|RATE1500|RATE2125|RATE2500|RATE3000|
RATE3125|RATE4250|RATE5000|RATE6000|RATE6250}
BUS:B<x>:S8B10B:BITRate ?
```

Related Commands BUS:B<x>:S8B10B:BITRate:VALue

Arguments

CUSTOM sets the bus data rate to Custom, allowing you to set the value (see Related Commands)

RATE1250 sets the bus data rate to 1.25 Gb/s

RATE1500 sets the bus data rate to 1.5 Gb/s.

RATE2125 sets the bus data rate to 2.125 Gb/s.

RATE2500 sets the bus data rate to 2.5 Gb/s.

RATE3000 sets the bus data rate to 3.0 Gb/s.

RATE3125 sets the bus data rate to 3.125 Gb/s.

RATE4250 sets the bus data rate to 4.25 Gb/s.

RATE5000 sets the bus data rate to 5.0 Gb/s.

RATE6000 sets the bus data rate to 6.0 Gb/s.

RATE6250 sets the bus data rate to 6.25 Gb/s.

Examples

BUS:B1:S8B10B:BITRATE 1500 sets the bit rate for the 8B10B bus B1 to 1.5 gigabits per second.

BUS:B1:S8B10B:BITRATE? might return :BUS:B1:S8B10B:BITRATE 1.25000E+9, indicating that the bit rate for the 8B10B bus B1 is set to 1.25 gigabits per second.

BUS:B<x>:S8B10B:BITRate:VALue

Sets the data rate for the specified bus to a rate that you specify in bits per second. The bitrate must be set to custom (see Related Commands, below).

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:S8B10B:BITRate:VALue <NR3>
BUS:B<x>:S8B10B:BITRate:VALue?
```

Related Commands BUS:B<x>:S8B10B:BITRate

Arguments <NR3> is the data rate in bits per second.

Examples `BUS:B2:S8B10B:BITRATE:VALUE 1.2500E+9` sets the data rate for 8B10B bus B2 to 1.25 Gb/s.

`BUS:B2:S8B10B:BITRATE:VALUE?` might return `:BUS:B<x>:S8B10B:BITRATE:VALUE 1.25e9`, indicating that the data rate for bus B2 is set to 1.25 gigabits per second.

BUS:B<x>:S8B10B:HYSTeresis

Sets or queries the hysteresis for the specified bus.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:S8B10B:HYSTeresis <NR3>
BUS:B<x>:S8B10B:HYSTeresis?
```

Arguments <NR3> is the hysteresis setting.

Examples `BUS:B1:S8B10B:BITRATE:VALUE 20e-3` sets the B1 bus value to 20 mV.

`BUS:B2:S8B10B:BITRATE:VALUE?` might return `:BUS:B2:S8B10B:BITRATE:VALUE 20.000e-3`, indicating that the B2 bus value is set to 20 mV.

BUS:B<x>:S8B10B:SOUrce

Sets or queries the signal sources for the specified bus.

NOTE. *Not available on some models.*

Group Bus

Syntax

```
BUS:B<x>:S8B10B:SOUrce {CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|
D5|D6|D7|D8|D9|D10|D11|D12|D13|D14|D15|
MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:S8B10B:SOUrce?
```

Arguments CH1–CH4 specifies an analog channel to use as the bus signal source.

D0–D15 specifies a digital channel to use as the bus signal source.

MATH1–MATH4 specifies a math channel to use as the bus signal source.

Examples BUS:B1:S8B10B:SOURCE CH2 sets the bus B1 signal source to channel 2.

BUS:B1:S8B10B:SOURCE? might return :BUS:B1:S8B10B:SOURCE D5, indicating that the bus B1 signal source is digital channel 5.

BUS:B<x>:SPI:BITOrder

NOTE. *Not available on some models.*

This command sets or queries the shift direction used by the SPI buss decoding operation to deserialize the data bit sequence into recovered word values.

Group Bus

Syntax

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

Related Commands BUS:B<x>:SPI:DATa:SIZe

Arguments LSB specifies that each bit becomes the recovered value's new LSB, after shifting previously recovered bits one place to the left. The decoding happens right to left.

MSB specifies that each successive bit from the bus's data line becomes the new MSB of the recovered value, shifting any previously recovered bits one place to the right. The decoding happens left to right.

Examples

`BUS:B1:SPI:BITORDER LSB` sets each bit order for the SPI bus B1 to LSB.

`BUS:B1:SPI:BITORDER?` might return `:BUS:B1:SPI:BITORDER MSB`, indicating that the bit order for the SPI bus B1 is set to MSB.

BUS:B<x>:SPI:CLOCk:POLarity

NOTE. *Not available on some models.*

This command sets or queries the SPI clock (SCK) polarity for the specified bus.

Group

Bus

Syntax

```
BUS:B<x>:SPI:CLOCk:POLarity {FALL|RISE}
BUS:B<x>:SPI:CLOCk:POLarity?
```

Related Commands

BUS:B<x>:SPI:CLOCk:SOUrce

Arguments

`FALL` sets the clock to the falling edge of the signal.

`RISE` sets the clock to the rising edge of the signal.

Examples

`BUS:B1:SPI:CLOCK:POLARITY FALL` sets the SPI clock polarity to the falling edge for the bus B1.

`BUS:B1:SPI:CLOCK:POLARITY?` might return `:BUS:B1:SPI:CLOCK:POLARITY RISE`, indicating that the SPI clock polarity for the bus B1 is set to the rising edge.

BUS:B<x>:SPI:CLOCk:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the SPI clock (SCK) source for the specified bus.

Group

Bus

Syntax

```
BUS:B<x>:SPI:CLOCk:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:SPI:CLOCk:SOUrce?
```

Related Commands BUS:B<x>:SPI:CLOCk:POLarity

Arguments `D0–D15` designates a digital input signal as the clock source.

`CH1–CH4` designates an analog channel as the buss SPI clock source.

`MATH1–MATH4` designates a math waveform as the clock source.

Examples `BUS:B1:SPI:CLOCK:SOURCE D8` sets the SPI clock source for the bus B1 to D8.

`BUS:B1:SPI:CLOCK:SOURCE?` might return `:BUS:B1:SPI:CLOCK:SOURCE D11`, indicating that the SPI clock source for the bus B1 is set to D11.

BUS:B<x>:SPI:DATa:POLarity

NOTE. *Not available on some models.*

This command sets or queries the SPI data (DATA) polarity for the bus specified by x. The value of x can range from 1 through 16. The SPI decode operation treats high inputs as ones in normal polarity and zeros in inverted polarity.

Group Bus

Syntax

```
BUS:B<x>:SPI:DATa:POLarity {high|low}
BUS:B<x>:SPI:DATa:POLarity?
```

Related Commands

Arguments `high` sets the SPI data polarity to active high.

`low` sets the SPI data polarity to active low.

Examples `BUS:B1:SPI:DATA:POLARITY high` sets the SPI data polarity for the bus B1 to active high.

`BUS:B1:SPI:DATA:POLARITY?` might return `:BUS:B1:SPI:DATA:POLARITY LOW`, indicating that the SPI data polarity for the bus B1 is set to active low.

BUS:B<x>:SPI:DATa:SIZe

NOTE. Not available on some models.

This command sets or queries the number of bits per word for the specified SPI bus.

Group Bus

Syntax
```
BUS:B<x>:SPI:DATa:SIZe <NR3>
BUS:B<x>:SPI:DATa:SIZe?
```

Related Commands

Arguments <NR3> is the data size for the specified bus. The minimum value is 2 and maximum is 64.

Examples BUS:B1:SPI:DATA:SIZE 7 sets the data size for SPI bus B1 to seven bits per word.

BUS:B1:SPI:DATA:SIZE? might return :BUS:B1:SPI:DATA:SIZE 8, indicating that the data size for SPI bus B1 is set to eight bits per word.

BUS:B<x>:SPI:DATa:SOUrce

NOTE. Not available on some models.

This command sets or queries the SPI data (DATA) source for the bus specified by x. The value of x can range from 1 through 16.

Group Bus

Syntax
```
BUS:B<x>:SPI:DATa:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|
D11|D12|D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:SPI:DATa:SOUrce?
```

Arguments CH1–CH4 designates an analog channel as the data source for the specified SPI bus.

MATH1–MATH4 designates a math waveform as the data source.

D0–D15 designates a digital input signal as the data source.

Examples BUS:B1:SPI:DATA:SOURCE D2 sets the SPI data source for the bus B1 to D2.

BUS:B1:SPI:DATA:SOURCE? might return :BUS:B1:SPI:DATA:SOURCE D3, indicating that the SPI data source for the bus B1 is set to D3.

BUS:B<x>:SPI:FRAMING

This command sets or queries the SPI framing setting. The bus is specified by x.

NOTE. *Not available on some models.*

Group Bus

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

Arguments IDLE specifies IDLE SPI framing.

SS specifies SS SPI framing.

Examples BUS:B1:SPI:FRAMING IDLE sets the SPI framing to IDLE.

BUS:B1:SPI:FRAMING? might return :BUS:B1:SPI:FRAMING SS, indicating that the SPI framing is set to SS.

BUS:B<x>:SPI:IDLETime

This command sets or queries the SPI idle time. The bus is specified by x.

NOTE. *Not available on some models.*

Group Bus

Syntax
```
BUS:B<x>:SPI:IDLETime <nr3>
BUS:B<x>:SPI:IDLETime?
```

Arguments <nr3> specifies the SPI idle time.

Examples

`BUS:B1:SPI:IDLETime 0.000004` sets the idle time to 4 µs.

`BUS:B1:SPI:IDLETime?` might return `:BUS:B1:SPI:IDLETIME 5.0000E-6`, indicating that the idle time is set to 5 µs.

BUS:B<x>:SPI:SELect:POLarity

NOTE. *Not available on some models.*

This command sets or queries the SPI Slave Select (SS) polarity for the specified bus.

Group

Bus

Syntax

```
BUS:B<x>:SPI:SELect:POLarity {LOW|HIGH}
BUS:B<x>:SPI:SELect:POLarity?
```

Related Commands

BUS:B<x>:SPI:SELect:SOUrce

Arguments

`LOW` sets an active low polarity.

`HIGH` sets an active high polarity.

Examples

`BUS:B1:SPI:SELECT:POLARITY HIGH` sets the SPI Slave Select polarity for the bus B1 to active high.

`BUS:B1:SPI:SELECT:POLARITY?` might return `:BUS:B1:SPI:SELECT:POLARITY HIGH`, indicating that the SPI Slave Select polarity for the bus B1 is set to active high.

BUS:B<x>:SPI:SELect:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the SPI Slave Select (SS) source for the specified bus.

Group

Bus

Syntax

```
BUS:B<x>:SPI:SELect:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:SPI:SELect:SOUrce?
```

Related Commands BUS:B<x>:SPI:SELect:POLarity

Arguments `CH1–CH4` designates an analog channel as the busses' SPI Slave Select source.

`D0–D15` designates a digital input signal as the Slave Select source.

`MATH1–MAThH4` designates a math waveform as the Slave Select source.

Examples `BUS:B1:SPI:SELECT:SOURCE MATH5` sets the SPI Slave Select source for the bus B1 to MATH5.

`BUS:B1:SPI:SELECT:SOURCE?` might return `:BUS:B1:SPI:SELECT:SOURCE D3`, indicating that the SPI Slave Select source for the bus B1 is set to D3.

BUS:B<x>:TYPe

NOTE. *Not available on some models.*

This command sets or queries the type for the specified bus.

Group Bus

Syntax

```
BUS:B<x>:TYPe
{CAN|CUSTom|FLEXRAY|LIN|I2C|MIPICSITWo|MIPIDSIOne|
PARallel|PCIE|RS232c|S8B10B|SPI|USB|ETHernet}
BUS:B<x>:TYPe?
```

Arguments `CAN` specifies a CAN bus.

`CUSTom` specifies a custom bus.

`FLEXRAY` specifies a FLEXRAY bus.

`LIN` specifies a LIN bus.

`I2C` specifies the Inter-IC bus.

`MIPICSITWo` specifies the MIPI CSI2 bus.

`MIPIDSIOne` specifies the MIPI DSI1 bus.

PARallel specifies the Parallel bus.

PCIE specifies a PCIe bus.

RS232 specifies the RS232 Serial bus.

S8B10B specifies the 8B10B bus.

SPI specifies the Serial Peripheral Interface bus.

USB specifies the Universal Serial Bus.

ETHernet specifies the Ethernet bus.

Examples

BUS:B1:TYPE I2C sets the bus B1 type to I2C.

BUS:B1:TYPE? might return :BUS:B1:TYPE SPI, indicating that the bus B1 type is set to SPI and that the bus behavior is governed by the :BUS:B1:SPI:XXXX commands.

BUS:B<x>:USB:BITRate

NOTE. *Not available on some models.*

This command sets or queries the USB bit rate for bus <x>, where x is the bus number.

Group

Bus

Syntax

```
BUS:B<x>:USB:BITRate {FULL|HIGH|LOW}
BUS:B<x>:USB:BITRate?
```

Arguments

FULL indicates the bit rate is 12 Mbps.

HIGH indicates the bit rate is 480 Mbps.

LOW indicates the bit rate is 1.5 Mbps.

Examples

BUS:B1:USB:BITRATE FULL sets the bit rate to 12 Mbps.

BUS:B1:USB:BITRATE? might return :BUS:B1:USB:BITRATE LOW, indicating that the bit rate is 1.5 Mbps.

BUS:B<x>:USB:PRObe

NOTE. *Not available on some models.*

This command sets or queries the type of probe connected to the USB signal for bus <x>, where x is the bus number.

Group Bus

Syntax

```
BUS:B<x>:USB:PRObe {DIFFerential|SINGleended}
BUS:B<x>:USB:PRObe?
```

Arguments `DIFFerential` indicates that the bus probe is a differential probe.

`SINGleended` indicates that the bus probe is a single-ended probe.

Examples `BUS:B1:USB:PROBE DIFFERENTIAL` sets the bus probe to be a differential probe.

`BUS:B1:USB:PROBE?` might return `:BUS:B1:USB:PROBE SINGLEENDED`, indicating that a single-ended probe is used.

BUS:B<x>:USB:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the USB Data Source for bus <x>, where x is the bus number.

Group Bus

Syntax

```
BUS:B<x>:USB:SOUrce
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:USB:SOUrce?
```

Related Commands BUS:B<x>:USB:SOUrce:DMINus, BUS:B<x>:USB:SOUrce:DPLUs

Arguments

CH1–Ch4 specifies an analog channel as the data source for the specified USB bus.

D0–D15 specifies a digital channel as the data source for the specified USB bus.

MATH1–MATH4 specifies a math channel as the data source for the specified USB bus.

Examples

BUS:B1:USB:SOURCE D2 sets the USB Data Source for bus B1 to D2.

BUS:B1:USB:SOURCE? might return :BUS:B1:USB:SOURCE D3, indicating that the USB Data Source for bus B1 is D3.

BUS:B<x>:USB:SOUrce:DMINus

NOTE. *Not available on some models.*

This command sets or queries the USB Data Source for D- input for bus <x>, where x is the bus number.

If you are using single-ended probes, you need to set the sources for both the D+ and D- inputs.

Group

Bus

Syntax

```
BUS:B<x>:USB:SOUrce:DMINus
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:USB:SOUrce:DMINus?
```

Related Commands

BUS:B<x>:USB:SOUrce, BUS:B<x>:USB:SOUrce:DPLUs

Arguments

CH1–Ch4 specifies an analog channel as the D- source for the specified USB bus.

D0–D15 specifies a digital channel as the D-source for the specified USB bus.

MATH1–MATH4 specifies a math channel as the D- source for the specified USB bus.

Examples

BUS:B1:USB:SOURCE:DMINUS CH2 sets the USB Data Source for D- input to CH2.

BUS:B1:USB:SOURCE:DMINUS? might return :BUS:B1:USB:SOURCE:DMINUS CH4, indicating that CH4 is set to be the D- input for USB data.

BUS:B<x>:USB:SOUrce:DPLUs

NOTE. *Not available on some models.*

This command sets or queries the USB Data Source for D+ input. If you are using single-ended probes, you need to set the sources for both the D+ and D- inputs.

Group Bus

Syntax

```
BUS:B<x>:USB:SOUrce:DPLUs
{CH1|CH2|CH3|CH4|D0|D1|D2|D3|D4|D5|D6|D7|D8|D9|D10|D11|D12|
D13|D14|D15|MATH1|MATH2|MATH3|MATH4}
BUS:B<x>:USB:SOUrce:DPLUs?
```

Related Commands BUS:B<x>:USB:SOUrce, BUS:B<x>:USB:SOUrce:DMINus

Arguments

`CH1–Ch4` specifies an analog channel as the D+ source for the specified USB bus

`D0–D15` specifies a digital channel as the D+ source for the specified USB bus.

`MATH1–MATH4` specifies a math channel as the D+ source for the specified USB bus.

Examples

`BUS:B1:USB:SOURCE:DPLUS D2` sets the USB Data Source for D+ input to D2.

`BUS:B1:USB:SOURCE:DPLUS?` might return `:BUS:B1:USB:SOURCE:DPLUS D3`, indicating that D3 is set to be the D+ input for USB data.

BUS:CH<x>:LOWTHRESHold

This command sets or queries the low threshold value for the analog source in the bus (USB differential).

Group Bus

Syntax

```
BUS:CH<x>:LOWTHRESHold <NR3>
BUS:CH<x>:LOWTHRESHold?
```

Related Commands BUS:CH<x>:THRESHold, BUS:MATH<x>:LOWTHRESHold, BUS:MATH<x>:THRESHold, BUS:REF<x>:THRESHold

Arguments <NR3> is a floating point number that specifies the threshold in millivolts. The default value is -200 mV.

Examples BUS:CH1:LOWTHRESHOLD -100.0000E-3 sets the low threshold value of the analog source to -100 mV.

BUS:CH1:LOWTHRESHOLD? might return :BUS:CH1:LOWTHRESHOLD: -100.0000E-3, indicating that the low threshold value for CH1 of the bus is -100 mV.

BUS:CH<x>:THRESHold

If there is a high and low threshold for the analog source in the bus, this command sets or queries the high threshold value. Otherwise, this command sets or queries the threshold value.

Group Bus

Syntax
```
BUS:CH<x>:THRESHold <NR3>
BUS:CH<x>:THRESHold?
```

Related Commands BUS:CH<x>:LOWTHRESHold, BUS:MATH<x>:LOWTHRESHold, BUS:MATH<x>:THRESHold, BUS:REF<x>:THRESHold

Arguments <NR3> is a floating point number that specifies the high threshold value for the specified analog source. The default value is 1.4 V.

Examples BUS:CH1:THRESHOLD 1.2500 sets the high threshold value to 1.25 V if there is a high and low threshold for the analog source for the bus; otherwise, sets the threshold value to 1.25 V.

BUS:CH1:THRESHOLD? might return :BUS:CH1:THRESHOLD 1.5000, indicating that the (high if applicable) threshold value for the analog source is 1.5 V.

BUS:MATH<x>:LOWTHRESHold

This command sets or queries the low threshold value of the mathematical waveform for the bus (USB differential).

Group Bus

Syntax

```
BUS:MATH<x>:LOWTHRESHold <NR3>
BUS:MATH<x>:LOWTHRESHold?
```

Related Commands BUS:MATH<x>:THRESHold, BUS:CH<x>:LOWTHRESHold, BUS:CH<x>:THRESHold, BUS:REF<x>:THRESHold

Arguments <NR3> is a floating point number that specifies the threshold in Volts. The default value is 0V.

Examples BUS:MATH1:LOWTHRESHOLD 1.000 sets the low threshold value of the mathematical waveform to 1V.

BUS:MATH1:LOWTHRESHOLD? might return :BUS:MATH1:LOWTHRESHOLD 100.0000E-3, indicating that the low threshold value of the mathematical waveform is 0.1 volts.

BUS:MATH<x>:THRESHold

If there is a high and low threshold for the mathematical waveform in the bus, this command sets or queries the high threshold value. Otherwise, this command sets or queries the threshold value.

Group Bus

Syntax

```
BUS:MATH<x>:THRESHold <NR3>
BUS:MATH<x>:THRESHold?
```

Related Commands BUS:MATH<x>:LOWTHRESHold, BUS:CH<x>:LOWTHRESHold, BUS:CH<x>:THRESHold, BUS:REF<x>:THRESHold

Arguments <NR3> is a floating point number that specifies the high threshold value for the mathematical waveform. The default value is 0 V.

Examples BUS:MATH2:THRESHOLD 1 sets the threshold value of the mathematical waveform to 1 volt.

BUS:MATH1:THRESHOLD? might return :BUS:MATH1:THRESHOLD 1.0000, indicating that the threshold value of the mathematical waveform is 1 volt.

BUS:REF<x>:THRESHold

If there is a high and low threshold for the reference waveform in the bus, this command sets or queries the high threshold value. Otherwise, this command sets or queries the threshold value.

Group Bus

Syntax

```
BUS:REF<x>:THRESHold <NR3>
BUS:REF<x>:THRESHold?
```

Related Commands BUS:CH<x>:LOWTHRESHold, BUS:CH<x>:THRESHold, BUS:MATH<x>:LOWTHRESHoldBUS:MATH<x>:THRESHold

Arguments <NR3> is a floating point number that specifies the high threshold value for the reference waveform. The default value is 0 V.

Examples BUS:REF1:THRESHOLD 1.5 sets the high threshold value for REF 1 to 1.5 V if there is a high and low threshold; otherwise, sets the threshold value to 1.5 V.

BUS:REF1:THRESHOLD? might return :BUS:REF1:THRESHOLD 1.5000, indicating that the REF 1 threshold is set to 1.5 volts.

BUSY? (Query Only)

This query-only command returns the status of the instrument. This command allows you to synchronize the operation of the instrument with your application program.

Group Status and Error

Syntax BUSY?

Related Commands *OPC, *WAI

Returns <NR1> = 0 means that the instrument is not busy processing a command whose execution time is extensive.

<NR1> = 1 means that the instrument is busy processing a command Commands that Generate an OPC Message (See Table 2-47.).

Examples BUSY? might return :BUSY 1, indicating that the instrument is currently busy.

*CAL? (Query Only)

This query-only command starts signal path compensation and returns the status upon completion.

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes. Any remote command that performs an action on the oscilloscope is also disabled until the SPC is complete.*

Group Calibration

Syntax *CAL?

Related Commands CALibrate:RESults:SPC?

Returns 0 indicates SPC passed.

-1 indicates SPC failed.

3 indicates the instrument is still in the warm up phase and SPC was not run.

Examples *CAL? starts the internal signal path calibration and returns the status upon completion.

CALibrate? (Query Only)

This query returns the status of instrument calibration, including internal signal path calibration, factory calibration, and probe calibration.

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes.*

Group Calibration

Syntax CALibrate?

Examples `CALIBRATE?` might return `:CALIBRATE:INTERNAL:STATUS PASS;CALIBRATE:FACTORY:STATUS PASS0;:CALIBRATE:RESULTS:SPC PASS;:CALIBRATE:PROBESTATE:CH1 1;CH2 1;CH3 1;CH4 1`, indicating the calibration status.

CALibrate:CALProbe:CH<x>? (Query Only)

This query-only command instructs the instrument to perform a probe calibration for the selected channel and returns the calibration status. The Channel <x> range is 1 through 4. This command is equivalent to selecting Probe Cal from the Vertical menu. You must warm up the instrument for at least 20 minutes before running this command.

NOTE. *A probe query takes approximately 30 seconds to complete. No other commands will be executed until calibration is complete.*

Group Calibration

Syntax `CALibrate:CALProbe:CH<x>?`

Related Commands CALibrate:PRObestate:CH<x>?, CH<x>:PROBECal?

Returns `-1` indicating the probe calibration failed for the selected channel.

`0` indicating the probe calibration passed for the selected channel.

`1` indicating the probe calibration is initialized for the selected channel.

Examples `CALIBRATE:CALPROBE:CH2?` might return `:CALIBRATE:CALPROBE:CH2 0`, indicating the probe calibration passed for Channel 2.

CALibrate:INTERNal (No Query Form)

This command (no query form) starts the internal signal path calibration (SPC) of the instrument. You can use the CALibrate:INTERNal:STATus? query to return the current status of the internal signal path calibration of the instrument.

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes.*

Group Calibration

Syntax `CALibrate:INTERNal`

Related Commands CALibrate:INTERNal:STATus?, CALibrate:RESults:SPC?

Examples `CALIBRATE:INTERNAL` starts the internal signal path calibration of the instrument.

CALibrate:INTERNal:STARt (No Query Form)

This command (no query form) starts the internal signal path calibration (SPC) of the instrument. This command is the same as the CALibrate:INTERNal command. You can use the CALibrate:INTERNal:STATus? query to return the current status of the internal signal path calibration of the instrument.

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes.*

Group Calibration

Syntax `CALibrate:INTERNal:STARt`

Related Commands CALibrate:INTERNal:STATus?, CALibrate:RESults:SPC?

Examples `CALIBRATE:INTERNAL:START` starts the internal signal path calibration of the instrument.

CALibrate:INTERNal:STATus? (Query Only)

This query-only command returns the current status of the internal signal path calibration.

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes.*

Group Calibration

Syntax `CALibrate:INTERNal:STATus?`

Related Commands *CAL?, CALibrate:RESults:SPC?

Returns This query will return one of the following:

- `INIT` indicates the instrument has not had internal signal path calibration run.
- `WARMUP` indicates the instrument has not warmed up for the required time to perform signal path calibration.
- `PASS` indicates that the signal path calibration completed successfully.
- `FAIL` indicates that the signal path calibration did not complete successfully.
- `TEMPDRIFT` indicates that the instrument has not reached a state of stable temperature adequate for performing signal path calibration.
- `RUNNING` indicates that 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:PRObestate:CH<x>? (Query Only)

This query-only command returns the probe calibration status for the probe of the selected channel, 1 through 4.

Group Calibration

Syntax `CALibrate:PRObestate:CH<x>?`

Related Commands CH<x>:PROBECal?

Returns This query will return one of the following:

- –1 Indicating the probe calibration failed for the selected channel.
- 0 Indicating the probe calibration passed for the selected channel.
- 1 Indicating the probe calibration is initialized for the selected channel.
- 2 Indicating that the probe calibration for the selected channel is running.

Examples `CALIBRATE:PROBESTATE:CH1?` might return `:CALIBRATE:PROBESTATE:CH1 0`, indicating that the probe calibration for Channel 1 passed.

CALibrate:RESults? (Query Only)

This query-only command 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 instrument calibration readiness: all returned results should indicate PASS status if the instrument is "fit for duty". It is quite common, however, to use uncalibrated probes (particularly when the instrument inputs are connected into a test system with coaxial cables).

This command is equivalent to selecting Instrument Calibration from the Utilities menu and then viewing the contents of the Status field.

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes.*

Group Calibration

Syntax `CALibrate:RESults?`

Related Commands *CAL?

Examples `CALIBRATE:RESULTS?` returns the status of all the calibration subsystems. The query might return `:CALIBRATE:RESULTS:SPC PASS;FACTORY PASS`, indicating that the status of internal and factory calibrations, without performing any calibration operations is set to PASS.

CALibrate:RESults:SPC? (Query Only)

This query-only command returns the results of the last SPC operation. However, this query does not cause an SPC to be run. This command is equivalent to selecting Instrument Calibration from the Utilities menu.

NOTE. *When running SPC through the remote interface, calibration status cannot be obtained until after the SPC completes, which can take several minutes.*

Group Calibration

Syntax CALibrate:RESults:SPC?

Related Commands *CAL?

Examples CALIBRATE:RESULTS:SPC? returns the results of the last SPC operation: either PASS, FAIL, WARMUP, TEMPDRIFT, INIT, or RUNNING.

CHANNELMAPping? (Query Only)

This query returns a list of all channel mappings (such as, CH1.B2,CH2.A2,CH3.C2,CH4.D2). DPO70000SX Series only.

Group Miscellaneous

Syntax CHANNELMAPping?

Examples CHANNELMAPPING? may return the channel mapping CH1.B2,CH2.A2,CH3.C2,CH4.D2.

CH<x>? (Query Only)

This query-only command returns the vertical parameters for the specified channel. The channel is specified by x. The value of x can range from 1 through 4. This command is similar to selecting Vertical Setup from the Vertical menu.

Group Vertical

Syntax CH<x>?

Examples CH1? might return the following vertical parameters for channel 1: :CH1:BANDWIDTH 1.0000E+09;COUPLING DC;DESKEW0.0000E+00;OFFSET 0.0000E+00;POSITION 0.0000E+00;SCALE 5.0000E-01;PROBCAL INIT;PROBE:GAIN 1.0000E+00;RESISTANCE 1.0000E+06;UNITS "V";ID:TYPE "1X";SERNUMBER "N/A";:CH1:PROBEFUNC:EXTATTEN 1.0000E+00;EXTUNITS "None";:CH1:LABEL:NAME "";XPOS 5;YPOS 5.

CH<x>:ATIACTive? (Query Only)

This query-only command returns whether the specified channel is an ATI channel.

Group Vertical

Syntax `CH<x>:ATIACTive?`

Examples `CH2:ATIACTive?` might return `:CH2:ATIACTIVE 1`, indicating that channel 2 is an ATI channel.

CH<x>:AVAILable? (Query Only)

This query-only command returns whether the specified channel is available.

Group Vertical

Syntax `CH<x>:AVAILable?`

Examples CH4:AVAILable? might return `:CH4:AVAILABLE 0`, indicating that channel 4 is not available.

CH<x>:BANdwidth

This command sets or queries the selectable low-pass bandwidth limit filter of the specified channel. The channel is specified by x. The value of x can range from 1 through 4. This is equivalent to selecting Bandwidth from the Vertical menu.

The query form of this command always returns the approximate realized bandwidth of the channel.

Available arguments depend upon the instrument and the attached accessories.

Group Vertical

Syntax
```
CH<x>:BANdwidth {<NR3>|FIVe|FULl|TWEnty|ONEfifty|TWOfifty}
CH<x>:BANdwidth?
```

Arguments

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

FIVe sets the upper bandwidth limit to 500 MHz.

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

TWEnty sets the upper bandwidth limit to 20 MHz.

ONEfifty sets the upper bandwidth limit to 150 MHz.

TWOfifty sets the upper bandwidth limit to 250 MHz.

Examples

CH1:BANDWIDTH TWENTY sets the bandwidth of Channel 1 to 20 MHz.

CH2:BANDWIDTH?, for DPO7054 instruments, might return :CH2:BANDWIDTH 500.0000E+06, indicating that there is no bandwidth limiting on Channel 2.

For DPO7104 instruments, this query might return :CH2:BANDWIDTH 1.0000E+09 indicating that there is no bandwidth limiting on Channel 2.

CH<x>:BANdwidth:ENHanced

This command sets or queries the "Bandwidth Enhancement (DSP) Enabled" or "Analog Only" option.

NOTE. *When possible, use the CH<x>:BANdwidth command to choose the enhanced bandwidth DSP filter.*

Group

Vertical

Syntax

```
CH<x>:BANdwidth:ENHanced {AUTO|OFF}
CH<x>:BANdwidth:ENHanced?
```

Arguments

AUTO allows "Bandwidth Enhanced (DSP) Enabled", filtering as possible.

OFF allows "Analog Only", preventing DSP filtering.

Examples

CH1:BANDWIDTH:ENHANCED AUTO allows bandwidth enhancement filtering as possible.

CH1:BANDWIDTH:ENHANCED? might return :CH1:BANDWIDTH:ENHANCED AUTO, indicating that the bandwidth will be enhanced if possible.

CH<x>:BANdwidth:ENHanced:APPLYtoall

This command sets or queries specified channel's bandwidth and bandwidth enhanced selection, “DSP” or “Analog Only” to set the other channels the same. Bandwidth achieved on each channel can further be limited while conditions, such as a lower bandwidth probe, that limited the bandwidth exist. A query will almost certainly return 0, as once the request is processed, ApplyToAll will return to its default state of OFF allowing the channels to continue operating independently.

Group Vertical

Syntax

```
CH<x>:BANdwidth:ENHanced:APPLYtoall {ON|OFF|1|0}
CH<x>:BANdwidth:ENHanced:APPLYtoall?
```

Arguments

`ON` sets other channels to bandwidth and bandwidth enhancement filters or analog only of the specified channel.

`OFF` is the default setting, each channel operates independently regarding bandwidth and bandwidth enhanced (DSP).

`1` sets other channels to bandwidth and bandwidth enhancement filters or analog only of the specified channel.

`0` is the default and each channel operates independently regarding bandwidth and bandwidth enhanced (DSP).

Examples

`CH1:BANDWIDTH:ENHANCED:APPLYTOALL ON` sets the CH1 Bandwidth selection and Digital Filters (DSP) Enabled or Analog Only selection to the other 3 channels. The bandwidth set is influenced by the BW limitations of the channel. If CH1 is not physically limited and has requested 15.0 GHz (DSP) bandwidth, while CH2 has a P7313 probe attached which is limited to 12.5 GHz (HW). A request via CH1 ApplyToAll might result in 13 GHz (DSP) on CH2. If no other Bandwidth selections were made on CH2, subsequent removal of the probe on CH2 would result in 15 GHz (DSP) on CH2.

`CH:BANDWIDTH:ENHANCED:APPLYTOALL?` might return `:CH:BANDWIDTH:ENHANCED:APPLYTOALL 0`, indicating that each channel operates independently regarding bandwidth and bandwidth enhanced (DSP).

CH<x>:BANdwidth:ENHanced:FORCe

This command sets or queries the Force constant sample rate of the Digital Filters (DSP). The bandwidth Enhanced Filter provides you the ability to “Force”

constant sample rate, keeping the system in the sample rate base. This is also known as Bandwidth Enhanced AUTO.

NOTE. *The bandwidth set is influenced by the requested bandwidth and its channel limitations.*

NOTE. *If Force is ON, a request to lower the sample rate below the DSP supported rate is rejected.*

Group Vertical

Syntax
```
CH<x>:BANdwidth:ENHanced:FORCe {OFF|ON}
CH<x>:BANdwidth:ENHanced:FORCe?
```

Arguments `OFF` is the default setting; DSP filtering occurs when sample rate is supported.

`ON` forces the conditions appropriate for Digital Filtering (DSP) to occur. With this option system changes are made to achieve sample rate that Bandwidth Enhanced Digital Filters (DSP) operate in, a Horizontal Mode of Constant Sample Rate, appropriate DSP BW, and Bandwidth Enhanced Auto on the specified channel would all be selected. Incompatible features would be turned off for example, FastAcq would be inhibited.

Examples `CH1:BANDWIDTH:ENHANCED:FORCE?` might return `:CH1:BANDWIDTH:ENHANCED:FORCE ON`, request to lower the sample rate below the DSP supported rate would be rejected.

CH<x>:BANdwidth:ENHanced:STATE? (Query Only)

This query-only command returns the state of the bandwidth enhancement filter (DSP).

Group Vertical

Syntax
```
CH<x>:BANdwidth:ENHanced:STATE?
```

Examples `CH1:BANDWIDTH:ENHANCED:STATE?` might return `:CH1:BANDWIDTH:ENHANCED:STATE 1`, indicating the bandwidth enhancement filter is on.

CH<x>:COUPling

This command sets or queries the input attenuator coupling setting for the specified channel. The channel is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Coupling from the Vertical menu.

NOTE. *The available arguments depend on the instrument and the attached accessories.*

Group Vertical

Syntax

```
CH<x>:COUPling {AC|DC|DCREJect|GND}
CH<x>:COUPling?
```

Arguments

AC sets the specified channel to AC coupling.

DC sets the specified channel to DC coupling.

DCREJect sets DC Reject coupling when probes are attached that have that feature.

GND sets the specified channel 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

This command sets or queries the deskew time for the specified channel. The channel is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Deskew from the Vertical Setup menu.

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.

Group Vertical

Syntax

```
CH<x>:DESKew <NR3>
CH<x>:DESKew?
```

Arguments

<NR3> is the deskew time for this channel, ranging from -25 ns to +25 ns with a resolution of 1 ps. Out-of-range values are clipped.

Examples

CH4:DESKEW 5.0E-9 sets the deskew time for Channel 4 to 5 ns.

CH2:DESKEW? might return :CH2:DESKEW 2.0000E-09, indicating that the deskew time for Channel 2 is set to 2 ns.

CH<x>:FASTAcqcapable? (Query Only)

This query-only command returns whether the specified channel is capable of FastAcq.

Group

Vertical

Syntax

```
CH<x>:FASTAcqcapable?
```

Examples

CH2:FASTAcqcapable? might return :CH2:FASTACQCAPABLE 0, indicating that channel 2 is not FastAcq capable.

CH<x>:FASTFRamecapable? (Query Only)

This query-only command returns whether the specified channel is capable of FastFrame acquisitions.

Group

Vertical

Syntax

```
CH<x>:FASTFRamecapable?
```

Examples

CH2:FASTFRamecapable? might return :CH2:FASTFRAMECAPABLE 0, indicating that channel 2 is not capable of FastFrame.

CH<x>:ICAPture:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the digital source to acquire iCapture signals.

NOTE. *To use this command,* `iCapture` *should be turned on.*

Group Vertical

Syntax

```
CH<x>:ICAPture:SOUrce {D0|D1|D2|D3|D4|D5|D6|D7
|D8|D9|D10|D11|D12|D13|D14|D15|CQ0}
CH<x>:ICAPture:SOUrce?
```

Related Commands CH<x>:ICAPture:STATE

Arguments `D0–D15` or `CQ0` specifies the digital channel.

Examples `CH1:ICAPTURE:SOURCE D8` selects the signal from digital probe input D8 as the signal CH1 acquires in iCapture mode (`:CH1:ICAPture:STATE is ON`).

`CH1:ICAPTURE:SOURCE?` might return `:CH1:ICAPTURE:SOURCE D9`, indicating that the signal from digital probe input is set to D9.

CH<x>:ICAPture:STATE

NOTE. *Not available on some models.*

This command sets or queries the state of the iCapture feature for the channel specified by x. The value of x can range from 1 through 4.

Group Vertical

Syntax

```
CH<x>:ICAPture:STATE {<NR1>|OFF|ON}
CH<x>:ICAPture:STATE?
```

Related Commands CH<x>:ICAPture:SOUrce

Arguments `<NR1>` = 0 turns off the iCapture feature; any other value turns on the iCapture feature.

`OFF` turns off the iCapture feature.

ON turns on the iCapture feature.

Examples

CH1:ICAPTURE:STATE ON turns on the iCapture feature.

CH1:ICAPTURE:STATE? might return :CH1:ICAPTURE:STATE 0, indicating that the iCapture feature is turned off.

CH<x>:INVert

This command sets or queries the invert function for the specified channel. The channel is specified by x. The value of x can range from 1 through 4. When on, the invert function inverts the waveform for the specified channel.

Group

Vertical

Syntax

```
CH<x>:INVert {ON|OFF|<NR1>}
CH<x>:INVert?
```

Arguments

OFF argument turns off the invert function for the specified channel.

ON argument turns on the invert function for the specified channel.

<NR1> = 0 turns off the invert function; any other value turns on the invert function.

Examples

CH4:INVert ON inverts the waveform on channel 4.

CH2:INVert? might return :CH2:INVERT 0 indicating that the invert function for channel 2 is off.

CH<x>:LABel:NAMe

This command sets or queries the label attached to the displayed waveform for the specified channel. The channel is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Label from the Vertical menu.

Group

Vertical

Syntax

```
CH<x>:LABel:NAMe <QString>
CH<x>:LABel:NAMe?
```

Arguments <QString> is an alphanumeric character string, ranging from 1 through 32 characters in length.

Examples CH2:LABEL:NAME "Pressure" changes the waveform label for the Channel 2 waveform to “Pressure”.

CH3:LABEL:NAME? might return :CH3:LABEL:NAME "Force", indicating that the waveform label for the Channel 3 waveform is “Force”.

CH<x>:LABel:XPOS

This command sets or queries the X screen offset at which the label (attached to the displayed waveform of the specified channel) is displayed, relative to the left edge of the screen. The channel is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Label from the Vertical menu and either viewing or setting X Pos.

Group Vertical

Syntax
```
CH<x>:LABel:XPOS <NR1>
CH<x>:LABel:XPOS?
```

Arguments <NR1> is the location (control in divisions) where the waveform label for the selected channel is displayed, relative to the left edge of the screen. Arguments should be integers ranging from 0 through 10.

Examples CH3:LABEL:XPOS 5 moves the waveform label for the Channel 3 that it begins 5 divisions to the right of the left edge of the screen.

CH2:LABEL:XPOS? might return :CH2:LABEL:XPOS 5, indicating that the waveform label for the Channel 2 currently 5 divisions to the right of the left edge of the screen.

CH<x>:LABel:YPOS

This command sets or queries the Y screen offset at which the label (attached to the displayed waveform of the specified channel) is displayed, relative to the waveform handle. The channel is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Label from the Vertical menu and either viewing or setting Y Pos.

Group Vertical

Syntax

```
CH<x>:LABel:YPOS <NR1>
CH<x>:LABel:YPOS?
```

Arguments

<NR1> is the location (in divisions) where the waveform label for the selected channel is displayed, relative to the waveform handle. Arguments should be integers ranging from 10 to -10.

Examples

CH3:LABel:YPOS -2.5 moves the waveform label for the Channel 3 2.5 divisions below the waveform handle.

CH2:LABel:YPOS? might return :CH2:LABEL:YPOS 0, indicating that the waveform label for the Channel 2 is currently located at the waveform handle.

CH<x>:OFFSet

This command sets or queries the vertical offset for the specified channel. The channel is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Offset from the Vertical menu.

This command offsets the vertical acquisition window (moves the level at the vertical center of the acquisition window) for the selected 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 instrument always displays the input signal minus the offset value. The channel reference marker will move to the vertical graticule position given by the negative of the offset value divided by the scale factor, unless that position is off-screen. If the computed coordinate for the reference mark is off-screen, the mark moves to the nearest screen limit and changes from a right-pointing arrow (→) to an arrow pointing in the appropriate off-screen direction.

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.

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>:OPTIcal:RCVR

This command sets or queries the Optical Reference Receiver DSP filter used to compensate for the applied Optical signal.

Group Vertical

Syntax
```
CH<x>:OPTIcal:RCVR {OFF | FLAT | ENET257R4 | ENET266PAM4 |
OTU2795 | FC28050}
```

Arguments The Optical Reference Receiver DSP filter used to compensate for the applied Optical signal.

Examples `CH1:OPTIcal:RCVR ENET257R4` sets the DSP filter to ENET257R4.

`CH1:OPTIcal:RCVR?` might return `:CH1:OPTIcal:RCVR ENET257R4` indicating the DSP filter is set to ENET257R4.

CH<x>:OPTIcal:WLENgth

Get or set the optical wavelength (in nanometers) that is used by the Probe and oscilloscope to compensate for the applied Optical signal.

Group Vertical

Syntax
```
CH<x>:OPTIcal:WLENgth <NR1>
CH<x>:OPTIcal:WLENgth?
```

Arguments `<NR1>` is the optical wavelength (in nanometers).

CH<x>:OPTIcal:WLENgth:LIST? (Query Only)

Query returns the comma-separated list of wavelengths with calibrated responses from the Probe. The units for the returned values are in nanometers.

Group Vertical

Syntax `CH<x>:OPTIcal:WLENgth:LIST?`

CH<x>:POSition

This command sets or queries the vertical position of the specified channel. The channel is specified by x. The value of x can range from 1 through 4. The position value is applied to the signal before it is digitized. This command is equivalent to selecting Position/Scale from the Vertical menu and either viewing or setting Position.

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, whether it is a channel, math, or reference waveform. 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.

Group Vertical

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

Related Commands CH<x>:OFFSet, D<x>:POSition,MATH<x>:VERTical:POSition, REF<x>:VERTical:POSition,

Arguments <NR3> is the position value, in divisions from the center graticule, ranging from 8 to -8 divisions.

Examples `CH2:POSITION 1.3E+00` positions the Channel 2 input signal 1.3 divisions above the center graticule.

`CH1:POSITION?` might return `:CH1:POSITION -1.3000E+00`, indicating that the current position of Channel 1 is 1.3 divisions below the center graticule.

CH<x>:PRObe? (Query Only)

This query-only command returns all information concerning the probe that is attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4.

Group Vertical

Syntax `CH<x>:PRObe?`

Related Commands CH<x>:PROBECal?

Examples `CH2:PROBE?` might return `:CH2:PROBE:GAIN 1.0000E-01; RESISTANCE 1.0000E+07;UNITS "V";ID:TYPE "10X"'SERNUMBER "N/A"` for a 10X probe, indicating that (among other parameters) the attenuation factor for the probe attached to Channel 2 is 100.0 mV (assuming that probe units are set to volts).

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

This command sets the probe attached to the input to autozero. The channel is specified by x. The value of x can range from 1 through 4.

Group Vertical

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

Arguments `EXECute` sets the probe attached to the specified channel to autozero.

Examples `CH1:PROBE:AUTOZERO EXECUTE` sets the probe attached to the Channel 1 to autozero.

CH<x>:PRObe:DEGAUSS (No Query Form)

This command starts a degauss cycle of the probe attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4.

Group Vertical

Syntax `CH<x>:PRObe:DEGAUSS EXECute`

Arguments `EXECute` starts the degauss cycle.

Examples `CH1:PROBE:DEGAUSS EXECUTE` causes the probe attached to Channel 1 to degauss.

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

This command queries whether the probe attached to the specified channel is degaussed.

Group Vertical

Syntax `CH<x>:PRObe:DEGAUSS:STATE?`

Examples `CH2:PROBE:DEGAUSS:STATE?` might return `:CH2:PROBE:DEGAUSS:STATE PASSED`, indicating that the probe attached to the Channel 2 is degaussed.

CH<x>:PRObe:FORCEDRange

This command sets the attached probe to the specified range, or it queries the range of the probe attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4.

A partial list of probes supported by this command includes the following probes: TCP0001, TCP0030, TCP0150, TDP0500, TDP1000, and TDP1500.

- `CH<x>:PRObe:FORCEDRange` response will not appear in CH<x>:PRObe? or *LRN? query responses for channels that do not have probes that implement the appropriate features.
- This command can be sent to any channel whose 'range-selection' policy is Manual without regard to the kind of probe installed in that channel (an error results if the policy is Auto). This permits accepting previous SET? or *LRN? query responses even if the attached probe is different from the one when the query was sent.
- The `CH<x>:PRObe:FORCEDRange?` query can be sent explicitly to any channel even though these commands do not appear in a "SET?" response, but the oscilloscope will reply with the string "N/A" if the probe does not implement those control mechanisms.

Group Vertical

Syntax
```
CH<x>:PRObe:FORCEDRange <dynamicRangeNR3>
CH<x>:PRObe:FORCEDRange?
```

Arguments <dynamicRangeNR3> specifies the probe range.

Examples If a TCP0030 current probe is attached to the Channel 1 input, CH1:PROBE:FORCEDRANGE 5.0 sets the attached probe to its 5 Ampere range.

CH1:PROBE:FORCEDRANGE? might return :CH1:PROBE:FORCEDRANGE 2.5000, indicating that the range of the probe attached to the Channel 1 is set to 2.5 Ampere.

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

This query-only command returns the gain factor of the probe that is attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4. 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.

Group Vertical

Syntax
```
CH<x>:PRObe:GAIN?
```

Related Commands CH<x>:SCAle

Examples CH2:PROBE:GAIN? might return :CH2:PROBE:GAIN 0.1000E+00, indicating that the attached 10X probe delivers 0.1 V to the Channel 2 BNC for every 1.0 V applied to the probe input.

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

This query-only command returns the type and serial of the probe that is attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4.

Group Vertical

Syntax `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)

This query-only command returns the serial number of the probe that is attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4.

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

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)

This query-only command returns the type of probe that is attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4. Level 2 (or higher) probes supply their exact product nomenclature; for Level 0 or 1 probes, a generic 'type string' of "nnX" 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:INPUTMode

This command sets or queries the input mode of the probe that is attached to the specified channel.

Group Vertical

Syntax

```
CH<x>:PRObe:INPUTMode {DEFault|DIFFerential|COMmonmode|A|B}
CH<x>:PRObe:INPUTMode?
```

Arguments

`DEFault` sets to the default.

`DIFFerential` sets the probe to route differential signals to the host.

`COMmonmode` sets the probe to route common-mode signals to the host.

`A` sets the probe to route single-ended A signals to the host.

`B` sets the probe to route single-ended B signals to the host.

Examples

`CH1:PROBE:INPUTMODE 5.0` sets the attached probe to Channel 1 to its 5 Ampere range.

`CH1:PROBE:INPUTMODE?` might return `:CH1:PROBE:INPUTMODE A`, indicating that the input mode of the probe that is attached to the Channel 1 is set to DEFAULT

CH<x>:PRObe:INPUTMode:AOFFSet

This command sets or queries the requested A mode offset control of the probe that is attached to the specified channel.

Group Vertical

Syntax

```
CH<x>:PRObe:INPUTMode:AOFFSet <NR3>
CH<x>:PRObe:INPUTMode:AOFFSet?
```

Arguments

`<NR3>` specifies the A mode offset control.

Examples

`CH1:PROBE:INPUTMODE:AOFFSET 5.0` sets the requested A mode offset control of the probe that is attached to the Channel 1 to its 5 Ampere range.

CH1:PROBE:INPUTMODE:AOFFSET? might return :CH1:PROBE:INPUTMODE:AOFFSET? 0.0000, indicating that the requested A mode offset control of the probe that is attached to the Channel 1 is set to 0.0.

CH<x>:PRObe:INPUTMode:BOFFSet

This command sets or queries the requested B mode offset control of the probe that is attached to the specified channel.

Group Vertical

Syntax

```
CH<x>:PRObe:INPUTMode:BOFFSet <NR3>
CH<x>:PRObe:INPUTMode:BOFFSet?
```

Arguments <NR3> specifies the B mode offset control.

Examples CH1:PROBE:INPUTMODE:BOFFSET 5.0 sets the requested B mode offset control of the probe that is attached to the Channel 1 to its 5 Ampere range.

CH1:PROBE:INPUTMODE:BOFFSET? might return :CH1:PROBE:INPUTMODE:BOFFSET 0.0000, indicating that the requested A mode offset control of the probe that is attached to the Channel 1 is set to 0.0.

CH<x>:PRObe:INPUTMode:CMOFFSet

This command sets or queries the requested common mode offset control of the probe that is attached to the specified channel.

Group Vertical

Syntax

```
CH<x>:PRObe:INPUTMode:CMOFFSet <NR3>
CH<x>:PRObe:INPUTMode:CMOFFSet?
```

Arguments <NR3> specifies the common mode offset control.

Examples CH1:PROBE:INPUTMODE:CMOFFSET 5.0 sets the requested common mode offset control of the probe that is attached to the Channel 1 to its 5 Ampere range.

CH1:PROBE:INPUTMODE:CMOFFSET? might return :CH1:PROBE:INPUTMODE:CMOFFSET 0.0000, indicating that the requested

common mode offset control of the probe that is attached to the Channel 1 is set to 0.0.

CH<x>:PRObe:INPUTMode:DMOFFSet

This command sets or queries the requested common mode differential mode offset control of the probe that is attached to the specified channel.

Group Vertical

Syntax

```
CH<x>:PRObe:INPUTMode:DMOFFSet <NR3>
CH<x>:PRObe:INPUTMode:DMOFFSet?
```

Arguments <NR3> specifies the differential mode offset control.

Examples CH1:PROBE:INPUTMODE:DMOFFSET 5.0 sets the requested differential mode offset control of the probe that is attached to the Channel 1 to its 5 Ampere range.

CH1:PROBE:INPUTMODE:DMOFFSET? might return :CH1:PROBE:INPUTMODE:DMOFFSET 0.0000, indicating that the requested differential mode offset control of the probe that is attached to the Channel 1 is set to 0.0.

CH<x>:PRObe:RANge

This command controls or queries the attenuation range of the probe on the designated channel.

A partial list of probes supported by this command is: P7260, P7313, P7313SMA, P7340A, P7360, P7360A, P7380, P7380A, P7380SMA, and P7520.

- CH<x>:PRObe:RANge response will not appear in CH<x>:PRObe? or *LRN? query responses for channels that do not have probes that implement the appropriate features.
- This command can be sent to any channel whose 'range-selection' policy is Manual without regard to the kind of probe installed in that channel (an error results if the policy is Auto). This permits accepting previous SET? or *LRN? query responses even if the attached probe is different from the one when the query was sent.
- The CH<x>:PRObe:RANge? query can be sent explicitly to any channel even though these commands do not appear in a SET? response, but the oscilloscope will reply with the string "N/A" if the probe does not implement those control mechanisms.

Group Vertical

Syntax

```
CH<x>:PRObe:RANge <rangeName>
CH<x>:PRObe:RANge
{ATTEN1X|ATTEN1_25X|ATTEN1_5X|ATTEN1_75X|ATTEN2X|
ATTEN2_5X|ATTEN3X|ATTEN3_5X|ATTEN4X|ATTEN4_5X|
ATTEN5X|ATTEN5_5X|ATTEN6X|ATTEN6_5X|ATTEN7X|
ATTEN7_5X|ATTEN8X|ATTEN8_5X|ATTEN9X|ATTEN9_5X|
ATTEN10X|ATTEN12_5X|ATTEN15X|ATTEN17_5X|ATTEN20X|
ATTEN25X|ATTEN30X|ATTEN35X|ATTEN40X|ATTEN45X|
ATTEN50X|ATTEN55X|ATTEN60X|ATTEN65X|ATTEN70X|
ATTEN75X|ATTEN80X|ATTEN85X|ATTEN90X|ATTEN95X|
ATTEN100X|ATTEN125X|ATTEN150X|ATTEN175X|ATTEN200X|
ATTEN250X|ATTEN300X|ATTEN350X|ATTEN400X|ATTEN450X|
ATTEN500X|ATTEN550X|ATTEN600X|ATTEN650X|ATTEN700X|
ATTEN750X|ATTEN800X|ATTEN850X|ATTEN900X|ATTEN950X|
ATTEN1000X}
CH<x>:PRObe:RANge?
```

Examples `CH1:PROBE:RANGE ATTEN10X` sets the attenuation range of the probe attached to Channel 1 to 10X.

`CH1:PROBE:RANGE?` might return `:CH1:PROBE:RANGE ATTEN1_25X`, indicating that the attenuation range of the probe on the designated Channel 1 is set to 25X.

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

This query-only command returns the resistance factor of the probe that is attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4.

Group Vertical

Syntax

```
CH<x>:PRObe:RESistance?
```

Examples `CH2:PROBE:RESISTANCE?` might return `:CH2:PROBE:RESISTANCE 1.0000E+06`, indicating that the input resistance of the probe attached to Channel 2 is 1 MΩ.

CH<x>:PRObe:SET

This command sets or queries aspects of probe accessory user interfaces, for example probe attenuation factors. The available arguments for this command will vary depending on the accessory you attach to the instrument.

For the P7260 probe, you can select between two attenuation factors using either this GPIB command or the push switch on the probe. The probe enables the relevant path and adjusts the set of available vertical settings based on the characteristics of the path in use. The probe signal path selection is not kept in persistent storage. The probe will lose the selection if you reboot the instrument or remove the probe. Also, the instrument does not store the selection in the save/recall setup operation.

Group Vertical

Syntax

```
CH<x>:PRObe:SET <QString>
CH<x>:PRObe:SET?
```

Arguments <QString> is a quoted string representing a settable aspect of the attached accessory. Example strings for some probes are shown as follows:

ATTENuation 5X sets the P7260 probe to ±0.75 V dynamic range with 6 GHz bandwidth and 5X attenuation.

ATTENuation 25X sets the P7260 probe to ±3 V dynamic range with 6 GHz bandwidth and 25X attenuation.

VTERMsource AUTO sets the P7380SMA probe voltage termination source to auto.

VTERMsource INTernal sets the P7380SMA probe voltage termination source to internal.

VTERMsource EXTernal sets the P7380SMA probe voltage termination source to external.

Examples CH2:PROBE:SET "ATTEN 25X" sets the P7260 probe to the selection for ±3 V dynamic range with 3 GHz bandwidth and 25X attenuation.

CH2:PROBE:SET? might return :CH2:PROBE:SET "ATTENUATION 5X" indicating that the P7260 probe is set to the selection for ±0.75 V dynamic range with 6 GHz bandwidth and 5X attenuation.

CH2:PROBE:SET? "ATTEN" might return :CH2:PROBE:SET "ATTENUATION 25X", indicating that the P7260 probe is set to the selection for ±3 V dynamic range with 3 GHz bandwidth and 25X attenuation.

CH<x>:PRObe:SIGnal

This command sets or queries aspects of probe accessory user interfaces. The available arguments for this command will vary depending on the accessory you attach to the instrument.

Group Vertical

Syntax

```
CH<x>:PRObe:SIGnal {PASS|BYPass}
CH<x>:PRObe:SIGnal?
```

Arguments

`PASS` opens a relay passing your signal to the instrument.

`BYPass` closes a relay preventing your signal from reaching the instrument. During probe degauss, the signal should be bypassed.

Examples

`CH2:PROBE:SIGNAL PASS` opens a relay passing your signal on Channel 2 to the instrument.

`CH1:PROBE:SIGNAL?` might return `:CH1:PROBE:SIGNAL BYPASS`, indicating that the relay is closed preventing your Channel 1 signal from reaching the instrument.

CH<x>:PRObe:TIPtype

This command sets or queries the type of probe tip being used on the specified channel.

Group Vertical

Syntax

```
CH<x>:PRObe:TIPtype {HBWStraightflex|OTHer}
CH<x>:PRObe:TIPtype?
```

Arguments

`HBWStraightflex` lets the instrument know you are using a high bandwidth straight-flex probe tip.

`OTHer` lets the instrument know you are not using a high bandwidth straight-flex probe tip.

Examples

`CH1:PROBE:TIPTYPE HBWSTRAIGHTFLEX` sets the channel 1 probe tip type to HBWStraightflex.

CH1:PROBE:TIPTYPE? might return :CH1:PROBE:TIPTYPE OTHER, indicating the probe tip is not an HBWStraightflex type.

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

This query-only command returns a string describing the units of measure for the probe attached to the specified channel. The channel is specified by x. The value of x can range from 1 through 4.

Group Vertical

Syntax CH<x>:PRObe:UNIts?

Related Commands CH<x>:PROBEFunc:EXTUnits

Examples CH4:PROBE:UNITS? might return :CH4:PROBE:UNITS "V", indicating that the units of measure for the probe attached to Channel 4 are volts.

CH<x>:PROBECal? (Query Only)

This query-only command returns the probe calibration state for the selected channel. The channel is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Probe Cal from the Vertical menu.

Group Vertical

Syntax CH<x>:PROBECal?

Returns This query will return one of the following:

- FAIl signifies that the probe calibration has failed for the selected channel.
- INIT signifies that the probe calibration has not yet been run for the selected channel.
- PASS signifies the probe calibration has passed for the selected channel.

Examples CH2:PROBECAL? might return :CH2:PROBECAL PASS indicating that the probe calibration has passed for Channel 2.

CH<x>:PROBECOntrol

This command allows you to set or query the probe range.

Group Vertical

Syntax

```
CH<x>:PROBECOntrol {AUTO|MANual}
CH<x>:PROBECOntrol?
```

Arguments

AUTO sets the values. The probe range is automatically calculated.

MANual allows you to select various valid values for the probe connected to a particular channel.

Examples

CH2:PROBECONTROL AUTO sets the values and the probe range is automatically calculated.

CH2:PROBECONTROL? might return :CH2:PROBECONTROL MANUAL indicating that you can select various valid values for the probe connected to a particular channel.

CH<x>:PROBEFunc:EXTAtten

This command is used to specify the input-output ratio, of external attenuation or gain, between the signal and the instrument input channels.

The channel is specified by x. The value of x can range from 1 through 4.

The query form of this command returns the user-specified attenuation. Note that this command deals with an attenuation factor, not a gain factor, unlike CH<x>:PRObe? (This command returns a value independent of the external attenuation). For example, if you specify a 20x attenuation factor for channel 1, the commands return the following values (assuming that a 1x probe is presently attached, since the external attenuation is used in combination with the probe attenuation):

```
CH1:PRObe:EXTA? 20.00E+0

CH1:PRObe?  1.0E+0
```

This command is equivalent to selecting Attenuation from the Vertical menu, and then either viewing or setting Ext Atten.

Group Vertical

Syntax

```
CH<x>:PROBEFunc:EXTAtten <NR3>
CH<x>:PROBEFunc:EXTAtten?
```

Related Commands CH<x>:PROBEFunc:EXTDBatten

Arguments <NR3> is the attenuation value, which is specified as a multiplier in the range from 1.00E-10 to 1.00E+10.

Examples CH1:PROBEFUNC:EXTATTEN 167.00E-3 specifies an external attenuation, which is connected between the user's input signal and the input of the probe attached to Channel 1.

CH2:PROBEFUNC:EXTATTEN? might return :CH2:PROBEFUNC:EXTATTEN 1.0000E+00, indicating that the probe attached to Channel 2 is connected directly to the user's signal.

CH<x>:PROBEFunc:EXTDBatten

Use this command to specify the input-output ratio (expressed in decibel units) of external attenuation or gain between the signal and the instrument input channels.

The channel is specified by x. The value of x can range from 1 through 4.

The query form of this command returns the user-specified attenuation in decibels: 1X = 0 dB, 10X = 20 dB, 100X = 40 dB, and similar things.

This command is equivalent to selecting Attenuation from the Vertical menu, and then either viewing or setting Ext Att (dB).

Group Vertical

Syntax

```
CH<x>:PROBEFunc:EXTDBatten <NR3>
CH<x>:PROBEFunc:EXTDBatten?
```

Related Commands CH<x>:PROBEFunc:EXTAtten

Arguments <NR3> is the attenuation value, which is specified in the range from -200.00 dB to 200.00 dB.

Examples CH3:PROBEFUNC:EXTDBATTEN 2.5 specifies an external 2.5 dB attenuator on Channel 3.

CH1:PROBEFUNC:EXTDBATTEN? might return :CH1:PROBEFUNC:EXTDBATTEEN 2.5000E+00, indicating that the attenuation for Channel 1 is 2.5 dB.

CH<x>:PROBEFunc:EXTUnits

This command sets the unit of measurement for the external attenuator of the specified channel. The channel is specified by x. The value of x can range from 1 through 4. There is also a corresponding query that returns the user-specified unit of measurement for the external attenuator. Unless these units are set to the factory default string value of "None", they become the attenuated units of measurement for that channel. It is assumed that the probe connected to the specified channel is of the correct type to receive the output of the user's external transducer or network.

Group Vertical

Syntax

```
CH<x>:PROBEFunc:EXTUnits <QString>
CH<x>:PROBEFunc:EXTUnits?
```

Related Commands CH<x>:PRObe:UNIts?

Arguments <QString> can contain a string of up to eight characters to indicate the attenuation unit of measurement for the specified channel. However, most instrument attenuators only display the first two characters.

Examples CH4:PROBEFUNC:EXTUNITS "Pa" sets the unit of measurement for the Channel 4 external attenuator.

CH2:PROBEFUNC:EXTUNITS? might return :CH2:PROBEFUNC:EXTUNITS "Pa", indicating that the Channel 2 external attenuator units of measurement are Pascals.

CH<x>:SCAle

This command sets or queries the vertical scale of the specified channel. The channel is specified by x. The value of x can range from 1 through 4. Sending this command is equivalent to selecting Vertical Setup from the Vertical menu and then viewing or setting the Scale.

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 and 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<x>:VERTical:SCAle

Arguments <NR3> is the vertical channel scale in units per division.

Examples CH4:SCALE 100E-03 sets the Channel 4 scale to 100 mV per division.

CH2:SCALE? might return :CH2:SCALE 1.0000E+00, indicating that the current scale setting of Channel 2 is 1 V per division.

CH<x>:TERmination

This command sets the connected/disconnected status of a 50 Ω resistor, which can be connected between the specified channel's coupled input and instrument ground. The channel is specified by x. The value of x can range from 1 through 4. 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 equivalent to selecting Termination from the Vertical menu or toggling between termination values from the VERTICAL area of the front panel.

NOTE. *The available arguments depend on the instrument and the attached accessories.*

Group Vertical

Syntax

```
CH<x>:TERmination <NR3>
CH<x>:TERmination?
```

Arguments

<NR3> specifies the channel input resistance, which can be specified as 50 Ω or 1,000,000 Ω.

Examples

CH4:TERMINATION 50.0E+0 establishes 50 Ω impedance on Channel 1.

CH2:TERMINATION? might return :CH2:TERMINATION 50.0E+0, indicating that Channel 2 is set to 50 Ω impedance.

CH<x>:THRESHold

This command sets or queries the comparable threshold for converting the analog signal to digital form for the channel specified by x. The value of x can range from 1 through 4.

Group

Vertical

Syntax

```
CH<x>:THRESHold <NR3>
CH<x>:THRESHold?
```

Related Commands

CH<x>:POSition, CH<x>:SCAle

D<x>:THRESHold, MATH<x>:THRESHold

Arguments

<NR3> specifies the analog threshold in volts.

Examples

CH1:THRESHOLD 1 sets the threshold value of the channel CH1 to 1 volt.

CH1:THRESHOLD? might return CH1:THRESHOLD 500.0000E-3, indicating that the threshold value set for the channel CH1 is 0.5 volts.

CH<x>:VTERm:BIAS

This command sets or queries the termination voltage for a channel (if control is available). <x> is the channel.

Group

Vertical

Syntax

```
CH<x>:VTERm:BIAS <NR3>
CH<x>:VTERm:BIAS?
```

Arguments

<NR3> specifies the termination voltage.

Examples

CH4:VTERm:BIAS 1.5 sets the termination voltage on CH4 to 1.5 Volts.

CH2:VTERm:BIAS? might return CH2:VTERm:BIAS 0.0000, indicating that the termination voltage for CH2 is set to 0 volts.

CH<x>:VTERm:DUAL:A

This command sets or queries the termination voltage for probes with dual inputs that support settable termination voltage.

Group

Vertical

Syntax

```
CH<x>:VTERm:DUAL:A <NR3>
CH<x>:VTERm:DUAL:A?
```

Arguments

<NR3> specifies the termination voltage.

Examples

CH4:VTERm:DUAL:A 1.5 sets the termination voltage of the A input of the channel 4 probe to 1.5 Volts.

CH3:VTERm:DUAL:B? might return CH3:VTERM:DUAL:B 0.0000, indicating that the termination voltage for the B input of the probe plugged into the CH3 input is set to 0 volts.

CH<x>:VTERm:DUAL:B

This command sets or queries the termination voltage for probes with dual inputs that support settable termination voltage.

Group

Vertical

Syntax

```
CH<x>:VTERm:DUAL:B <NR3>
CH<x>:VTERm:DUAL:B?
```

Arguments

<NR3> specifies the termination voltage.

Examples `CH3:VTERm:DUAL:A 0.5` sets the termination voltage of the A input of the channel 3 probe to 0.5 Volts.

`CH3:VTERm:DUAL:B?` might return `CH3:VTERM:DUAL:B 0.0000`, indicating that the termination voltage for the B input of the probe plugged into the CH3 input is set to 0 volts.

CLEAR (No Query Form)

This command clears acquisitions, measurements, and waveforms.

Group Miscellaneous

Syntax `CLEAR {ALL}`

Arguments All clears acquisitions, measurements, and waveforms.

Examples `CLEAR ALL` clears all acquisitions, measurements, and waveforms.

*CLS (No Query Form)

This command (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. MAV 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 instrument status data structures.

CMDBatch

This command sets or queries the state of command batching. By batching commands, database transactions can be optimized, increasing command throughput. Also, batching allows for ALL commands in an individual batch to be order independent and accomplish the same result as if the commands were coupled.

The Batch state is persistent and will be saved across power cycles, but will not be saved and recalled as part of a setup. In a setup scenario, the factory initial value is enabled.

Group Miscellaneous

Syntax

```
CMDBatch {<NR1>OFF|ON}
CMDBatch?
```

Arguments <NR1> = 0 turns command batching off; any other value turns command batching on.

OFF turns command batching off.

ON turns command batching on.

Examples CMDBATCHOFF disables command batching.

CMDBATCH? might return :CMDBATCH 1, indicating that command batching is turned on.

COUnter (No Query Form)

This command is used for three separate purposes. DPO70000SX Series only. This command is optional.

First, the COUnter REFerence command takes the reference to be the last measured value when the command is received. After that, the displayed values are deltas from the reference value. In linear terms reference is just the offset.

Second, the COUnter CLEAR command zeroes both the measured and derived statistical values, and zeroes the number of acquisitions used to accumulate the statistics.

Third, the COUnter RESET command is only needed in the Trigger Source Frequency Application to recalculate the A->B Sequence Trig on Nth Event value. This is only necessary if the user has overridden the automatically calculated Trig on Nth Event value. The application automatically calculates the Trig on Nth Event value to achieve an acquisition update rate of about 1/3 second, to both maximize the number of digits in the measured result, and keep the acquisition rate lively.

Group Trigger

Syntax `COUnter <REFerence|CLEAR|RESet>`

Related Commands COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments `REFerence` takes the reference to be the last measured value when the command is received. After that, the displayed values are deltas from the reference value.

`CLEAR` zeroes both the measured and derived statistical values, and zeroes the number of acquisitions used to accumulate the statistics.

`RESet` recalculates the A->B Sequence Trig on Nth Event value.

Examples `COUNTER CLEAR` takes the reference to be the last measured value when the command is received.

`COUNTER REF` takes the reference to be the last measured value when the command is received. .

`COUNTER RESet` recalculates the A->B Sequence Trig on Nth Event value.

COUnter:DURation

Sets or queries the parameter that controls the schedule for updating the Log Table of the Trigger Source Frequency or A-B Trigger Time Interval Application Data when the COUnter:UPDate is TIME. The time interval between log updates is set to the number of seconds specified in the argument of this command. DPO70000SX Series only. This command is optional.

Group Trigger

Syntax
```
COUnter:DURation <nr1>
COUnter:DURation?
```

Related Commands COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments <nr1> is the time interval, in seconds, between log updates. The default is 60 seconds.

Examples COUNTER:DURATION 60 causes the log table to be updated every 30 seconds.

COUNTER:DURATION? query may return a value between 1 and 3600.

COUnter:GAIn

Sets or queries the Gain (scale) value for the Trigger Source Frequency or A-B Trigger Time Interval applications. When the time or frequency is measured, it is multiplied by the Gain before the value is displayed, used to calculate statistics, or logged in the table. Considered along with COUNT:REFerence <nr3>, this applies the linear relationship y = mx + b, where b is the Reference (or offset) and m is the gain (or scale). The default gain is 1.0. The gain may never be 0.0. DPO70000SX Series only. This command is optional.

Group Trigger

Syntax
```
COUnter:GAIn <nr2>
COUnter:GAIn?
```

Related Commands COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments <nr2> is the Gain (scale) value for the Trigger Source Frequency or A-B Trigger Time Interval applications

Examples COUNTER:GAIN 2.0 multiplies all counter measurements by 2.

COUNTER:GAIN? May return any positive real value greater than 0.0 and less than 1000.0.

COUnter:LOGAction (No Query Form)

This command clears all data from the Log Table of the Trigger Source Frequency or A-B Trigger Time Interval Application. DPO70000SX Series only. This command is optional.

Group Trigger

Syntax COUnter:LOGAction <CLEAR>

Related Commands COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments CLEAR empties the log table.

Examples COUNTER:LOGACTION CLEAR removes all data from the log table.

COUnter:LOGNumber

Sets or queries the parameter that controls the schedule for updating the Log Table of the Trigger Source Frequency or A-B Trigger Time Interval Application Data when the COUnter:UPDate is NUMBER. The number of acquisitions between log updates is set to the number specified in the argument of this command. DPO70000SX Series only. This command is optional.

Group Trigger

Syntax

```
COUnter:LOGNumber <nr1>
COUnter:LOGNumber?
```

Related Commands

COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments

<nr1> is he number of acquisitions between log updates. The default is 100 acquisitions.

Examples

COUNTER:LOGNUMBER 1000 set the number of acquisitions between log updates to 1000.

COUNTER:LOGNUMBER? query may return a value between 1 and 10000.

COUnter:LOGTable

Sets or queries the parameter that enables/disables the Log Table of the Trigger Source Frequency or A-B Trigger Time Interval Application Data. When disabled, no log is kept. DPO70000SX Series only. This command is optional.

Group

Trigger

Syntax

```
COUnter:LOGTable <OFF|ON>
COUnter:LOGTable?
```

Related Commands

COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments

OFF disables use of the log table.

ON enables use of the log table. The default is ON.

Examples

`COUNTER:LOGTABLE OFF` disables use of the log table.

`COUNTER:LOGTABLE?` query may return a value of OFF or ON.

COUnter:REFerence

Sets or queries the Reference (offset) value for the Trigger Source Frequency or A-B Trigger Time Interval applications. When the time or frequency is measured the offset is subtracted from the measured value before the value is displayed, used to calculate statistics, or logged in the table. Considered along with COUNT:GAIn <nr3>. this applies the linear relationship y = mx + b, where b is the Reference (or offset) and m is the gain (or slope). This command may be used to zero the reference value (COUnter:REFerence 0.0), or to set it to any other value. The Reference can be taken automatically as the last measured value using the COUnter REF command (notice no semicolon here), and then queried to determine its value. DPO70000SX Series only. This command is optional.

Group

Trigger

Syntax

```
COUnter:REFerence <nr3>
COUnter:REFerence?
```

Related Commands

COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments

<nr3> is the Reference (offset) value for the Trigger Source Frequency or A-B Trigger Time Interval applications.

Examples

`COUNTER:REFERENCE 0.0` sets the Reference (offset) value for the Trigger Source Frequency or A-B Trigger Time Interval applications to 0.0.

`COUNTER:REFERENCE?` may return any positive real value within the range of possible measurement values.

COUnter:RESULTs? (Query Only)

Queries the measured and derived values obtained from the Trigger Source Frequency or A-B Trigger Time Interval Applications. The measured quantity

is value. The minimum, maximum, AVGmean, and Deviation are statistical values accumulated over the number of acquisitions (NUMber) taken at the time the command is given. All results are returned with the single query COUnter:RESULTs? Individual results are returned by other queries

Group Trigger

Syntax `COUnter:RESULTs?`

Related Commands COUnter:STAteCOUnter:TYPeCOUnter:SHOWMeasurementCOUnter:VIEWCOUnter:REFerenceCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Examples `COUNTER:RESULTS?` may return the set of results “10.0000000005 GH”,”10.000000001 GHz”, “10.00000008 GHz”, “10.00000003 GHz”, “124.3 mHz”, “1687”.

COUnter:RESULTs:AVGmean? (Query Only)

Queries the derived value obtained from the Trigger Source Frequency or A-B Trigger Time Interval Applications. The AVGmean is a statistical value accumulated over the number of acquisitions (NUMber) taken at the time the command was given.

Group Trigger

Syntax `COUnter:RESULTs:AVGmean?`

Related Commands COUnter:STAteCOUnter:TYPeCOUnter:SHOWMeasurementCOUnter:VIEWCOUnter:REFerenceCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Examples `COUNTER:RESULTS:AVGMEAN?` may return the specific measured result “10.000000001 GHz”.

COUnter:RESULTs:DEViation? (Query Only)

Queries the derived value obtained from the Trigger Source Frequency or A-B Trigger Time Interval Applications. The Deviation is a statistical value accumulated over the number of acquisitions (NUMber) taken at the time the command was given.

Group Trigger

Syntax `COUnter:RESULTs:DEViation?`

Related Commands COUnter:STAteCOUnter:TYPeCOUnter:SHOWMeasurementCOUnter:VIEWCOUnter:REFerenceCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Examples `COUNTER:RESULTS:DEVIATION?`may return the specific measured result "10.000000001 GHz"

COUnter:RESULTs:MAXimum? (Query Only)

Queries the derived value obtained from the Trigger Source Frequency or A-B Trigger Time Interval Applications. The maximum is a statistical value accumulated over the number of acquisitions (NUMber) taken at the time the command was given.

Group Trigger

Syntax `COUnter:RESULTs:MAXimum?`

Related Commands COUnter:STAteCOUnter:TYPeCOUnter:SHOWMeasurementCOUnter:VIEWCOUnter:REFerenceCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Examples `COUNTER:RESULTS:MAXIMUM?` may return the statistical result "10.000000001 GHz"

COUnter:RESULTs:MINimum? (Query Only)

Queries the derived value obtained from the Trigger Source Frequency or A-B Trigger Time Interval Applications. The minimum is a statistical value accumulated over the number of acquisitions (“NUMber”) taken at the time the command was given.

Group Trigger

Syntax `COUnter:RESULTs:MINimum?`

Related Commands COUnter:STAteCOUnter:TYPeCOUnter:SHOWMeasurementCOUnter:VIEWCOUnter:REFerenceCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Examples `COUNTER:RESULTS:MINIMUM?` may return the statistical result “10.000000001 GHz”.

COUnter:RESULTs:NUMber? (Query Only)

Queries the derived value obtained from the Trigger Source Frequency or A-B Trigger Time Interval Applications. The value is the number of acquisitions (NUMber) taken at the time the command was given.

Group Trigger

Syntax `COUnter:RESULTs:NUMber?`

Related Commands COUnter:STAteCOUnter:TYPeCOUnter:SHOWMeasurementCOUnter:VIEWCOUnter:REFerenceCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Examples `COUNTER:RESULTS:NUMBER?` may return 1687 indicating the number of acquisitions used in the measurement.

COUnter:RESULTs:VALue? (Query Only)

Queries the measured value obtained from the Trigger Source Frequency or A-B Trigger Time Interval Applications. The measured quantity is value.

Group Trigger

Syntax

```
COUnter:RESULTs:VALue?
```

Related Commands COUnter:STAteCOUnter:TYPeCOUnter:SHOWMeasurementCOUnter:VIEWCOUnter:REFerenceCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Examples `COUNTER:RESULTS:VALUE?` may return the specific measured result "10.000000001 GHz".

COUnter:SHOWLog

Sets or queries the parameter that controls the display of the Trigger Source Frequency or A-B Trigger Time Interval Application Data. The display of the data in the graticule area may be turned OFF or back ON as desired. A possible reason for turning it OFF is when the Log Table is enabled to record the data. DPO70000SX Series only. This command is optional.

Group Trigger

Syntax

```
COUnter:SHOWLog <OFF|ON>
COUnter:SHOWLog?
```

Related Commands COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments

OFF disables the display of the frequency or time interval data.

ON enables the display of the frequency or time interval data. The default is ON.

Examples

COUNTER:SHOWLOG ON displays the Trigger Source Frequency or A-B Trigger Time Interval Application Data.

COUNTER:SHOWLOG? query may return OFF or ON.

COUnter:SHOWMeasurement

Sets or queries the parameter that controls the display of the Trigger Source Frequency or A-B Trigger Time Interval Application Data. The display of the data in the graticule area may be turned OFF or back ON as desired. A possible reason for turning it OFF is when the Log Table is enabled to record the data. DPO70000SX Series only. This command is optional.

Group

Trigger

Syntax

```
COUnter:SHOWMeasurement <OFF|ON>
COUnter:SHOWMeasurement?
```

Related Commands

COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments

OFF disables the display of the frequency or time interval data.

ON enables the display of the frequency or time interval data. The default is ON.

Examples

COUNTER:SHOWMEASUREMENT ON enables display of Trigger Source Frequency or A-B Trigger Time Interval application data.

COUNTER:SHOWMEASUREMENT? query may return OFF or ON.

COUnter:STAte

Enables the Trigger Source Frequency or A-B Trigger Time Interval application, depending on the value of the COUnter:TYPe parameter. This command sets or

queries the Counter State enumerated value (OFF/ON) used to enable/disable the display of frequency or time information in the graticule area or in the Results Table.

The Trigger Source Frequency Application uses trigger hardware, rather than the acquired waveform, to measure the frequency of the signal on the trigger source to a very high precision. Best results are obtained when the External Reference is tied between the instrument and the source of the trigger signal. As a result of using the trigger hardware, the measurement update rate is very fast, 1/3 second by default. An A-Event Edge to B-Event Edge A->B Sequence must be used. Statistics are derived from the measured values: Minimum, Maximum, Average (mean), and Standard Deviation. The number of measurements made so far is also included. Note that long record lengths are NOT needed for this application, because the measurements are made from the trigger hardware, not from the acquired waveform. A Reference value may be taken by command or entered manually. When the Reference is non-zero, the measured value and the derived statistics are shown or queried as delta values from the Reference.

The A-B Trigger Time Interval Application also uses trigger hardware to measure the time between the A-Event and the B-Event in an A-B Sequence trigger. The A- and B-Events may be any combination of trigger types: Edge, Glitch, Width, etc. A Reference value may be taken by command or entered manually. When the Reference value is non-zero, the measured value is displayed or queried as a delta from this Reference.

For example, if the COUnter:TYPe parameter is set to FREQuency, the COUnter:STAte ON command causes the display of the frequency of the triggered channel signal, which is assumed to be an infinitely repeating clock. Along with the frequency value, the minimum, maximum, average (mean), and standard deviation of the frequency are also displayed. Finally, a reference value is displayed. The reference is 0.0 by default, but may be captured as the last measured value at any time using the COUnter REF command. When the reference value is nonzero, the other values are deltas from the reference value. The displayed information is typical of products generally referred to by the industry as “requency Counters. At the user’s discretion, the COUnter:VIEW <PERiod | FREQuency> command can be used to view the information as the Frequency (Hz) or Period (seconds). The default view is Frequency.

As a second example, if the COUnter:TYPe parameter is set to TIME, the COUnter:STAte ON command causes the display of the Time interval from the Trigger A-Event to the Trigger B-Event. These events can be from any trigger types in an A->B Trigger Sequence. In this case only the Time and the Reference value are displayed. The COUnter REF command takes the Reference value as before, but in this case the units relate to Time (seconds).

DPO70000SX Series only. This command is optional.

Group Trigger

Syntax

```
COUnter:STAte {<NR1>|OFF|ON}
COUnter:STAte?
```

Related Commands

COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:VIEWCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Arguments

<NR1>, if 0 disables the counter function, if 1 enables the counter function.

OFF disables the counter function. The default value is OFF.

ON enables the counter function.

Examples

COUNTER:STATE on enables the counter to display the count time or frequency.

COUNTER:STATE? query may return OFF or ON.

COUnter:TYPe

Sets or queries the Counter Application type as Trigger Source Frequency or A-B Trigger Time Interval. The Trigger Source Frequency Application is commonly referred to as Frequency Counting, and measures the frequency of the trigger source over a large number of events to obtain a very accurate result. The A-B Trigger Time Interval Application measures the A-Event to B-Event in an A->B Sequence Trigger, over a single A->B Trigger Sequence. DPO70000SX Series only. This command is optional.

Group

Trigger

Syntax

```
COUnter:TYPe <TIME|FREQuency>
COUnter:TYPe?
```

Related Commands

COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:VIEWCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?

Arguments

`TIME` sets the application type to frequency.

`FREQuency` sets the application type to time interval. The default value is Frequency.

Examples

`COUNTER:TYPE FREQuency` sets the counter type to frequency.

`COUNTER:TYPE?` query may return FREQuency or TIME.

COUnter:UPDate

Sets or queries the parameter that controls the schedule for updating the Log Table of the Trigger Source Frequency or A-B Trigger Time Interval Application Data. The schedule for logging the data may be set to occur automatically, after a given time interval has elapsed, or after a given number of acquisitions have occurred. The default is AUTO. DPO70000SX Series only. This command is optional.

Group

Trigger

Syntax

```
COUnter:UPDate <AUTO|TIME|NUMBER>
COUnter:UPDate?
```

Related Commands

COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments

`AUTO` sets the data logging to automatic.

`TIME` sets the data logging to occur after a given time interval has elapsed.

`NUMBER` sets the data logging to occur after a given number of acquisitions have occurred.

Examples

`COUNTER:UPDATE AUTO` sets the log table update schedule to automatic.

`COUNTER:UPDATE?` query may return AUTO, TIME, or NUMBER.

COUnter:VIEW

Sets or queries the View of the Counter Trigger Source Frequency Application Data. When the COUnter:TYPe is FREQuency, the data may be viewed as Frequency (Hz) or Time (seconds). This command does not apply to the A-B Time Interval Application (COUnter:TYPE TIME). DPO70000SX Series only. This command is optional.

Group Trigger

Syntax

```
COUnter:VIEW <FREQuency|PERiod>
COUnter:VIEW?
```

Related Commands COUnterCOUnter:DURationCOUnter:GAInCOUnter:LOGActionCOUnter:LOGNumberCOUnter:LOGTableCOUnter:REFerenceCOUnter:SHOWLogCOUnter:SHOWMeasurementCOUnter:STAteCOUnter:TYPeCOUnter:UPDateCOUnter:RESULTs?COUnter:RESULTs:AVGmean?COUnter:RESULTs:DEViation?COUnter:RESULTs:MAXimum?COUnter:RESULTs:MINimum?COUnter:RESULTs:NUMber?COUnter:RESULTs:VALue?COUnter:VIEW

Arguments

`FREQuency` displays the count in Hertz.

`PERiod` displays the count in seconds.

Examples

`COUNTER:VIEW FREQuency` displays the count in Hertz.

`COUNTER:VIEW?` query may return FREQuency or PERiod.

CQ<x>:THRESHold

NOTE. *Not available on some models.*

This command sets or queries the comparable threshold for converting the clock/qualifier signal to its digital form, where x is zero.

Group Digital

Syntax

```
CQ<x>:THRESHold <NR3>
CQ<x>:THRESHold?
```

Arguments <NR3> specifies the channel/qualifier threshold in volts.

Examples CQ0:THRESHOLD 0.5 sets the threshold value of the channel/qualifier CQ0 to 0.5 volt.

CQ0:THRESHOLD? might return CQ0:THRESHOLD 1.4000, indicating that the threshold value set for the channel/qualifier CQ0 is 1.4 volts.

CURSor?

This query-only command returns all of the current cursor settings.

Group Cursor

Syntax CURSor?

Examples CURSOR? might return the following as the current cursor settings:

```
:CURSOR:STATE 1;FUNCTION VBARS;LINESTYLE SOLID;MODE
INDEPENDENT;SOURCE1 CH1; SOURCE2 CH1;SCREEN:STYLE
LINES; :CURSOR:VBARS:UNITS SECONDS;POSITION1
-1.6000E-6;POSITION2 1.6000E-6; :CURSOR:HBARS:POSITION1
300.0000E-3; POSITION2 -300.0000E-3;UNITS BASE;
:CURSOR:WAVEFORM:POSITION1 -1.6000E-6;POSITION2 1.6000E-6;
SOURCE2 CH1;UNITS BASE;STYLE LINE_X;:CURSOR:XY :PRODDELTA
99.1000E+36;PRODUCT1 99.1000E+36;PRODUCT2 99.1000E+36;
READOUT RECTANGULAR;RADIUS1 99.1000E+36;RADIUS2 99.1000E+36;
RATDELTA 99.1000E+36;RATIO1 99.1000E+36;RATIO2 99.1000E+36;
RDELTA 99.1000E+36;RECTX1 0.0000;RECTX2 0.0000;RECTY1
0.0000; RECTY2 0.0000;THDELTA 99.1000E+36;THETA1
99.1000E+36; THETA2 99.1000E+36;XDELTA 0.0000;YDELTA 0.0000.
```

CURSor:FUNCtion

This command sets or queries the cursor type. Sending this command is equivalent to selecting Cursor Type from the Cursors menu, and then choosing from the drop-down list.

Group Cursor

Syntax

```
CURSor:FUNCtion {OFF|HBArs|VBArs|SCREEN|WAVEform}
CURSor:FUNCtion?
```

Related Commands CURSor:STATE

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

`HBArs` specifies horizontal bar cursors, which measure in vertical units.

`VBArs` specifies vertical bar cursors, which measure in horizontal units.

`SCREEN` specifies both horizontal and vertical bar cursors, which measure in horizontal and vertical units specified by the Cursor 1 and Cursor 2 Sources. Use these cursors to measure anywhere in the waveform display area.

`WAVEform` specifies paired or split 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.

Examples `CURSOR:FUNCTIONVBArs` selects the vertical bar cursors type.

`CURSOR:FUNCTION?` might return `:CURSor:FUNCtion HBArs`, indicating that the horizontal bar cursors are currently selected.

CURSor:HBArs? (Query Only)

This query-only command 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)

This query-only command 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 voltage difference between the two cursors is 5.08 V.

CURSor:HBArs:POSITION<x>

This command sets or queries 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. This command is the equivalent to selecting Cursor Position from the Cursors menu, selecting the H Bars Cursor Type, and then viewing or editing the desired cursor position.

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? (Query Only)

This query-only command returns the units for the horizontal bar cursors.

This query always returns BASE, indicating that the units for the horizontal bar cursors are those of the base waveform to which they were assigned.

Group Cursor

Syntax `CURSor:HBArs:UNIts?`

Examples `CURSOR:HBARS:UNITS?` always returns `:CURSOR:HBARS:UNITS BASE` indicating that the units for the horizontal bar cursors are base.

CURSor:LINESTyle

This command sets or queries the cursors line style used when cursors are displayed.

Group Cursor

Syntax
```
CURSor:LINESTyle {DASHed|SDASHed|SOLID}
CURSor:LINESTyle?
```

Arguments `DASHed` displays the cursors as dashed lines.

`SDASHed` displays the cursors as solid-dashed lines.

`SOLID` displays the cursors as solid lines.

Examples `CURSOR:LINESTYLESOLID` displays the cursors as solid lines.

`CURSOR:LINESTYLE?` might return `:CURSOR:LINESTYLE DASHED`, indicating that the cursors are displayed as dashed lines.

CURSor:MODe

This command sets or queries whether the two cursors move together in unison or separately. This command is the equivalent of selecting Cursor Mode from the Cursors menu, and then choosing from the drop-down list.

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:SCREEN:STYle

This command sets or queries the cursor type for Screen mode.

Group

Cursor

Syntax

```
CURSor:SCREEN:STYle {LINE_X|LINES|X}
CURSor:SCREEN:STYle?
```

Arguments

LINE_X specifies the cursor style to be a line with superimposed X.

LINES specifies the cursor style to be a line.

X specifies the cursor style to be an X.

Examples

CURSOR:SCREEN:STYLE LINES sets the cursor style to lines.

CURSOR:SCREEN:STYLE? might return :CURSOR:SCREEN:STYLE X, indicating that the style for screen cursors is X.

CURSor:SCREEN:XPOSITION<x>

This command sets or queries the x position of the specified screen cursor. The cursor is specified by x, which can be 1 or 2. If the cursor is not specified, it defaults to cursor 1.

Group

Cursor

Syntax

```
CURSor:SCREEN:XPOSITION<x> <NR3>
CURSor:SCREEN:XPOSITION<x>?
```

Related Commands CURSor:SCREEN:YPOSITION<x>

Arguments <NR3> specifies the x position of the specified screen cursor.

Examples CURSOR:SCREEN:XPOSITION1 15.0E-3 sets the x position of the screen Cursor 1 to 15 mV.

CURSOR:SCREEN:XPOSITION2? might return :CURSOR:SCREEN:XPOSITION2 -64.0000E-03 indicating that x position of the screen Cursor 2 is set to -64 mV.

CURSor:SCREEN:YPOSITION<x>

This command sets or queries the y position of the specified screen cursor. The cursor is specified by x, which can be 1 or 2. If the cursor is not specified, it defaults to cursor 1.

Group Cursor

Syntax
```
CURSor:SCREEN:YPOSITION<x> <NR3>
CURSor:SCREEN:YPOSITION<x>?
```

Related Commands CURSor:SCREEN:XPOSITION<x>

Arguments <NR3> specifies the y position of the specified screen cursor.

Examples CURSOR:SCREEN:YPOSITION1 25.0E-3 sets the y position of the screen Cursor 1 to 25 mV.

CURSOR:SCREEN:YPOSITION2? might return :CURSOR:SCREEN:YPOSITION2 -53.0000E-03 indicating that y position of the screen Cursor 2 is set to -53 mV.

CURSor:SOUrce<x>

This command sets or queries the source(s) for the currently selected cursor type. The cursor is specified by x, which can be 1 or 2. If the cursor is not specified, it defaults to cursor 1.

This command is equivalent to selecting Cursor Setup from the Cursors menu, and then choosing the desired cursor source.

Group Cursor

Syntax

```
CURSor:SOUrce<x>
{CH1|CH2|CH3|CH4|MATH1|MATH2|MATH3|MATH4|REF1|REF1|REF3|REF4}
CURSor:SOUrce<x>?
```

Arguments

CH1–CH4 sets an input channel waveform as the cursor source. The valid channel waveform range is from 1 through 4.

MATH1–MATH4 sets a math waveform as the cursor source. The valid math waveform range is from 1 through 4.

REF1–REF4 sets a reference waveform as the cursor source. The valid reference waveform range is from 1 through 4.

Returns

Any valid waveform.

Examples

CURSOR:SOURCE 1 CH2 sets the Cursor1 source to Channel 2.

CURSOR:SOURCE2? might return :CURSOR:SOURCE2 CH2 indicating that the Cursor2 source is Channel 2.

CURSOR:SOURCE? might return :CURSOR:SOURCE1 CH2, indicating that the Cursor1 source is Channel 2. If you do not specify the cursor, the cursor 1 source is returned.

CURSor:STATE

This command sets or queries the state of cursors. Note that setting the cursor state does not modify the cursor type.

This command is equivalent to pressing the CURSOR button on the front panel.

Group

Cursor

Syntax

```
CURSor:STATE {<NR1>|ON|OFF}
CURSor:STATE?
```

Related Commands

CURSor:FUNCtion

Arguments

<NR1> = 0 disables the display cursors; any other value enables the display cursors.

ON displays the cursors.

OFF removes cursors from the display.

Examples

`CURSOR:STATE ON` displays the cursors.

`CURSOR:STATE?` might return `:CURSOR:STATE 1`, indicating that the cursors are active on the display.

CURSor:VBArs

This command sets or queries the current settings for the vertical bar cursors or forces them to snap to positions specified by the DATa:STARt and DATa:STOP commands.

Group

Cursor

Syntax

```
CURSor:VBArs SNAp
CURSor:VBArs?
```

Related Commands

DATa:STARt, DATa:STOP

Arguments

`SNAp` forces the position of vertical bar cursor 1 and 2 to snap to the waveform record points specified by the DATa:STARt and DATa:STOP commands, respectively.

Examples

`CURSOR:VBARS SNAp` forces vertical bar cursor1 to the DATA:START position and vertical bar cursor 2 to the DATA:STOP position.

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

CURSor:VBArs:DELTa? (Query Only)

This query-only command returns the horizontal difference between the two vertical bar cursors. The units are specified by the CURSor:VBArs:UNIts command. The position can appear in units of Base or 1/Base.

Group

Cursor

Syntax

```
CURSor:VBArs:DELTa?
```

Related Commands

CURSor:VBArs:UNIts

Returns A floating point value with an exponent.

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:POSITION<x>

This command sets or queries the horizontal position for vertical bar cursors. 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. The position can appear in units of base or 1/base.

This command is the equivalent of selecting Cursor Setup from the Cursors menu, selecting V Bars Cursor Type, and then viewing or editing the desired cursor position.

Group Cursor

Syntax

```
CURSor:VBArs:POSITION<x> <NR3>
CURSor:VBArs:POSITION<x>?
```

Related Commands CURSor:VBArs:UNIts

Arguments `<NR3>` specifies the cursor position.

Returns A floating point value with an exponent.

Examples `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 ms.

CURSor:VBArs:POS<x>

This command sets or queries the horizontal position for vertical bar cursors. 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. The position can appear in units of base or 1/base.

This command is the equivalent of selecting Cursor Setup from the Cursors menu, selecting V Bars Cursor Type, and then viewing or editing the desired cursor position.

Group Cursor

Syntax
```
CURSor:VBArs:POS<x> <NR3>
CURSor:VBArs:POS<x>?
```

Related Commands CURSor:VBArs:UNIts

Arguments <NR3> specifies the cursor position.

Returns A floating point value with an exponent.

Examples `CURSOR:VBARS:POS2 9.00E-6` positions the cursor2 vertical bar cursor at 9 ms.

`CURSOR:VBARS:POS?` this command might return `:CURSOR:VBARS:POS1 1.0000E-06`, indicating that the cursor1 vertical bar is positioned at 1 ms.

CURSor:VBArs:UNIts

This command sets or queries the units for the vertical bar cursors.

Group Cursor

Syntax
```
CURSor:VBArs:UNIts {SECOnds|HERtz}
CURSor:VBArs:UNIts?
```

Arguments `SECOnds` sets the units of the vertical bar cursors for the time domain (seconds).

`HERtz` sets the units of the vertical bar cursors for the frequency domain (Hertz).

Returns A string for `SECOnds` or `HERtz`, depending on the current vertical bar cursor units.

Examples `CURSOR:VBARS:UNITS HERTZ` sets the units of the vertical bar cursors to 1/seconds.

CURSOR:VBARS:UNITS? might return :CURSOR:VBARS:UNITS SECONDS, indicating that the units for the vertical bar cursor are currently set to seconds.

CURSor:WAVEform

This command sets or queries the current settings for the waveform cursors or forces them to snap to positions specified by the DATa:STARt and DATa:STOP commands.

Group Cursor

Syntax

```
CURSor:WAVEform SNAp
CURSor:WAVEform?
```

Related Commands DATa:STARt, DATa:STOP

Arguments SNAp forces the position of waveform cursor 1 and 2 to snap to the waveform record points specified by the DATa:STARt and DATa:STOP commands, respectively.

Examples CURSOR:WAVEFORM SNAP forces waveform cursor1 to the DATA:START position and waveform cursor 2 to the DATA:STOP position.

CURSOR:WAVEFORM? might return the following waveform cursor settings :CURSOR:WAVEFORM:UNITS SECONDS;POSITION1 1.0000E-06;POSITION2 9.0000E-06.

CURSor:WAVEform:HDELTA? (Query Only)

This query-only command returns the horizontal difference between the waveform cursors. This is the absolute value of the horizontal position of the first cursor minus the horizontal position of the second cursor. This is equivalent to the waveform delta-time readout value.

Group Cursor

Syntax CURSor:WAVEform:HDELTA?

Examples CURSOR:WAVEFORM:HDELTA? might return :CURSOR:WAVEFORM:HDELTA 1.03E-9, indicating that the time between the waveform cursors is 1.03 ns.

CURSor:WAVEform:HPOS<x>? (Query Only)

This query-only command returns the position of the specified waveform cursor. The cursor is specified by x, which can be 1 or 2. This corresponds to the v1 or v2 (cursor 1 or cursor 2) cursor readout.

Group Cursor

Syntax

```
CURSor:WAVEform:HPOS<x>?
```

Examples `CURSOR:WAVEFORM:HPOS1?` might return `:CURSOR:WAVEFORM:HPOS2 0.247344970703`, indicating that cursor 2 is at 247.3 mV with respect to ground on the source waveform.

CURSor:WAVEform:POSition<x>

This command sets or queries the position of a waveform cursor, which is constrained to be visible in the selected time base. (Use the CURSor:FUNCtion command to set the cursor function to Waveform.) The cursor is specified by x, which can be 1 or 2. This is equivalent to setting a value in the Position control in the Cursor1 or Cursor2 sections of the Cursor Setup dialog box (when Waveform is selected in the Function section).

Group Cursor

Syntax

```
CURSor:WAVEform:POSition<x> {<NR3>}
CURSor:WAVEform:POSition<x>?
```

Related Commands CURSor:FUNCtion

Arguments `<NR3>` specifies the cursor position measured relative to the time of the trigger point of the source waveform.

Examples `CURSOR:WAVEFORM:POSITION2 9.00E-6` sets the waveform cursor 2 position to 9 µs.

`CURSOR:WAVEFORM:POSITION2?` might return `:CURSOR:WAVEFORM:POSITION2 51.0701402806E-009`, indicating waveform cursor 2 is at 51.1 ns.

CURSor:WAVEform:STYle

This command sets or queries the cursor type for Waveform mode.

Group Cursor

Syntax

```
CURSor:WAVEform:STYle {LINE_X|LINES|X}
CURSor:WAVEform:STYle?
```

Arguments

`LINE_X` specifies the cursor style to be a line with superimposed X.

`LINES` specifies the cursor style to be a line.

`X` specifies the cursor style to be an X.

Examples

`CURSOR:WAVEFORM:STYLE LINES` sets the cursor style to lines.

`CURSOR:WAVEFORM:STYLE?` might return `:CURSOR:WAVEFORM:STYLE X`, indicating that the style for waveform cursors is X.

CURSor:WAVEform:UNIts

This command sets or queries the units for waveform cursors.

Group Cursor

Syntax

```
CURSor:WAVEform:UNIts BASe
CURSor:WAVEform:UNIts?
```

Arguments

`BASe` sets cursor units to base.

Examples

`CURSOR:WAVEFORM:UNITS BASe` sets the waveform cursor units to base.

`CURSOR:WAVEFORM:UNITS?` might return `:CURSOR:WAVEFORM:UNITS BASE`, indicating that the units for waveform cursors are base.

CURSor:WAVEform:VDELTA? (Query Only)

This query-only command returns the vertical difference between the waveform cursors. This is the absolute value of the vertical position of the first cursor minus

the vertical position of the second cursor. This is equivalent to the waveform delta-voltage readout value.

Group Cursor

Syntax `CURSor:WAVEform:VDELTA?`

Examples `CURSOR:WAVEFORM:VDELTA?` might return `:CURSOR:WAVEFORM:VDELTA 1.06E-3`, indicating that the voltage between the waveform cursors is 1.06 mV.

CURSor:XY? (Query Only)

This query-only command returns all of the XY cursor parameters.

NOTE. *The values returned for all XY cursor readouts, that do not pertain to the current* `CURSor:XY:READOUT` *setting, are returned as the IEEE representation of NAN (Not a Number): 99.1000E+36.*

Group Cursor

Syntax `CURSor:XY?`

Examples `CURSOR:XY?` might return `:CURSOR:XY:PRODDELTA 99.1000E+36;PRODUCT1` 99.1000E+36;PRODUCT2 99.1000E+36;READOUT RECTANGULAR;RADIUS1 99.1000E+36;RADIUS2 99.1000E+36;RATDELTA 99.1000E+36; RATIO 199.1000E+36;RATIO2 99.1000E+36;RDELTA 99.1000E+36;RECTX1 1.0000;RECTX2 1.0200;RECTY1 360.0001E-6;RECTY2360.0001E-6; THDELTA 99.1000E+36;THETA1 99.1000E+36;THETA2 99.1000E+36;XDELTA 20.0000E-3;YDELTA 0.0000.

CURSor:XY:PRODDELta? (Query Only)

This query-only command returns the product of the difference between the cursors X positions and Y positions when the `CURSor:XY:READOUT` is set to PRODuct. The ΔX × ΔY value is calculated as (Cursor 2 X - Cursor 1 X) × (Cursor 2 Y - Cursor 1 Y).

NOTE. *This query is only valid when the* CURSor:XY:READOUT *is set to PRODuct. If the readout is set other than product, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax CURSor:XY:PRODDELta?

Examples CURSOR:XY:PRODDELTA? might return :CURSOR:XY:PRODDELTA -24.9856, indicating that the ΔX × ΔY value is -24.9856 VV.

CURSor:XY:PRODUCT<x>? (Query Only)

This query-only command returns the product of the X and Y positions for the specified cursor when the CURSor:XY:READOUT is set to PRODuct. The cursor is specified by x, which can be 1 or 2.

NOTE. *This query is only valid when the* CURSor:XY:READOUT *is set to PRODuct. If the readout is set other than product, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax CURSor:XY:PRODUCT<x>?

Examples CURSOR:XY:PRODUCT1? might return :CURSOR:XY:PRODUCT1 806.4004E-3, indicating that the product of the X position times the Y position for Cursor 1 is 806.4004 mVV.

CURSor:XY:RADIUS<x>? (Query Only)

This query-only command returns the radius of the specified cursor when the CURSor:XY:READOUT is set to POLARCoord. The cursor is specified by x, which can be 1 or 2.

NOTE. *This query is only valid when the* CURSor:XY:READOUT *is set to POLARCoord. If the readout is set other than polar coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax `CURSor:XY:RADIUS<x>?`

Examples `CURSOR:XY:RADIUS2?` might return `:CURSOR:XY:RADIUS2 4.9600`, indicating that the Cursor 2 radius is 4.9600 V.

CURSor:XY:RATDELta? (Query Only)

This query-only command returns ratio of the difference between the cursors X position and Y position (ΔY ¸ ΔX) when the `CURSor:XY:READOUT` is set to RATio. The ratio is calculated as (Cursor 2 Y - Cursor 1 Y) ÷ (Cursor 2 X - Cursor 1 X).

NOTE. *This query is only valid when the `CURSor:XY:READOUT` is set to RATio. If the readout is set other than ratio, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax `CURSor:XY:RATDELta?`

Examples `CURSOR:XY:RATDELTA?` might return `:CURSOR:XY:RATDELTA -1.0167`, indicating that the ΔY/ΔX ratio is -1.0167.

CURSor:XY:RATIO<x>? (Query Only)

This query-only command returns ratio the of the X (horizontal) and Y (vertical) position for the specified cursor when the `CURSor:XY:READOUT` is set to RATio. The cursor is specified by x, which can be 1 or 2. The ratio is calculated as Y ÷ X.

NOTE. *This query is only valid when the `CURSor:XY:READOUT` is set to RATio. If the readout is set other than ratio, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax `CURSor:XY:RATIO<x>?`

Examples `CURSOR:XY:RATIO1?` might return `:CURSOR:XY:RATIO1 15.8730E-3` indicating that the ratio of the X and Y positions for Cursor 1 is 15.8730E-3.

CURSor:XY:RDELta? (Query Only)

This query-only command returns the delta radius (Δr) value when the `CURSOR:XY:READOUT` is set to POLARCoord.

NOTE. *This query is only valid when the* `CURSOR:XY:READOUT` *is set to POLARCoord. If the readout is set other than polar coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax `CURSor:XY:RDELta?`

Examples `CURSOR:XY:RDELTA?` might return `:CURSOR:XY:RDELTA 6.9600`, indicating that the difference between the cursors radius is 6.9600 V.

CURSor:XY:READOUT

This command sets or queries the XY cursor readout (available when `DISplay:FORMat` is set to XY).

Group Cursor

Syntax
```
CURSor:XY:READOUT {RATio|RECTangular|POLARCoord|PRODuct}
CURSor:XY:READOUT?
```

Related Commands DISplay:FORMat

Arguments `POLARCoord` displays the following values: r1, r2, Δr, q1, q2, Δq, t1, t2, Δt.

`PRODuct` displays the following values: X1 x Y1, X2 xY2, ΔX x ΔY, t1, t2, ΔDt.

`RATio` displays the following values: Y1 ¸ X1, Y2 ¸ X2, ΔY ¸ x ΔX, t1, t2, Δt.

`RECTangular` displays the following values: X1, X2, ΔX, Y1, Y2, ΔY, t1, t2, Δt.

Examples `CURSOR:XY:READOUT RECTangular` sets the XY cursor readout to rectangular.

`CURSOR:XY:READOUT?` might return `:CURSOR:XY:READOUT RECTANGULAR`, indicating that the XY cursor readout is set to rectangular mode.

CURSor:XY:RECTX<x>

This command sets or queries the X cursor position in rectangular coordinates. The cursor is specified by x, which can be 1 or 2.

NOTE. *This query is only valid when the* `CURSor:XY:READOUT` *is set to RECTangular. If the readout is set other than rectangular coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax
```
CURSor:XY:RECTX<x> {<NR3>}
CURSor:XY:RECTX<x>?
```

Arguments `<NR3>` is the X position of the specified cursor in rectangular coordinates.

Examples `CURSOR:XY:RECTX2 5.04` sets the X position of Cursor 2 to 5.04 V in rectangular coordinates.

`CURSOR:XY:RECTX1?` might return `:CURSOR:XY:RECTX1 5.1200`, indicating that the X position of Cursor 1 is set to 5.1200 in rectangular coordinates.

CURSor:XY:RECTY<x>

This command sets or queries the Y cursor position in rectangular coordinates. The cursor is specified by x, which can be 1 or 2.

NOTE. *This query is only valid when the* `CURSor:XY:READOUT` *is set to RECTangular. If the readout is set to other than rectangular coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax
```
CURSor:XY:RECTY<x> {<NR3>}
CURSor:XY:RECTY<x>?
```

Arguments <NR3> is the Y position of the specified cursor in rectangular coordinates.

Examples CURSOR:XY:RECTY1 2.5 sets the Y position of Cursor 1 to 2.5 V.

CURSOR:XY:RECTY2? might return :CURSOR:XY:RECTY2 4.8800, indicating that the position of Cursor 2 in rectangular coordinates is 4.88 V.

CURSor:XY:THDELta? (Query Only)

This query-only command returns the XY cursor angle delta in polar coordinates.

NOTE. *This query is only valid when the CURSor:XY:READOUT is set to POLARCoord. If the readout is set other than polar coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax CURSor:XY:THDELta?

Examples CURSOR:XY:THDELTA? might return :CURSOR:XY:THDELTA 136.3748, indicating that the Dq XY cursor angle is 136.3748 degrees.

CURSor:XY:THETA<x>? (Query Only)

This query-only command returns the XY cursor angle in polar coordinates. The cursor is specified by x, which can be 1 or 2.

NOTE. *This query is only valid when the CURSor:XY:READOUT is set to POLARCoord. If the readout is set other than polar coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax CURSor:XY:THETA<x>?

Examples CURSOR:XY:THETA2? might return :CURSOR:XY:THETA2 -89.0906, indicating that the Channel 2 XY cursor angle in polar coordinates is -89.0906 degrees.

CURSor:XY:XDELta? (Query Only)

This query-only command returns the XY cursor X delta value in rectangular coordinates.

NOTE. *This query is only valid when the* `CURSor:XY:READOUT` *is set to RECTangular. If the readout is set to other than rectangular coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax `CURSor:XY:XDELta?`

Examples `CURSOR:XY:XDELTA?` might return `:CURSOR:XY:XDELTA 40.0000E-3`, indicating that the DX value is 40 mV.

CURSor:XY:YDELta? (Query Only)

This query-only command returns the XY cursor Y delta value in rectangular coordinates.

NOTE. *This query is only valid when the* `CURSor:XY:READOUT` *is set to RECTangular. If the readout is set to other than rectangular coordinates, the IEEE representation of NAN (Not a Number) is returned: 99.1000E+36.*

Group Cursor

Syntax `CURSor:XY:YDELta?`

Examples `CURSOR:XY:YDELTA?` might return `:CURSOR:XY:YDELTA 40.0000E-6`, indicating that the DY value is 40 µV.

CURVe

This command transfers waveform data to and from the instrument. Each waveform that is transferred has an associated waveform preamble that contains information such as data format and scale.

The CURVe? query transfers data from the instrument. 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 CURVe command transfers waveform data to the instrument. 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.

Group

Waveform Transfer

Syntax

```
CURVe
CURVe {<Block>|<asc curve>}
CURVe?
```

Related Commands

DATa:DESTination, DATa:SOUrce, DATa:STARt, DATa:STOP, SAVe:WAVEform, SAVe:WAVEform:FILEFormat, WFMInpre?, WFMInpre:BYT_Nr, WFMOutpre?, WFMInpre:NR_FR?, WFMOutpre:NR_FR?

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)

NOTE. *<x> is hexadecimal format. The letters A-F denote several y bytes between 10 and 15 digits.*

<yyy> is the number of bytes to transfer. If width is 1, then all bytes on the bus are single data points. If width is 2, then all bytes on the bus are 2-byte pairs. If width is 4, then all bytes on the bus are 4-byte pairs. Use the WFMInpre:BYT_Nr command to set the width.

<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.

Examples

CURVE <Block> sets the format of the waveform data, transferred to and from the instrument, to binary format.

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 instrument asynchronously and, depending upon the length of the curve record, such transfers can require several seconds to complete. During this time, the instrument will not respond to user controls. You can interrupt these asynchronous data transfers by sending a device clear message to the instrument or by interrupting the query with another command or query. Verify that curve data is completely transferred.*

*It is recommended that you follow such queries with an *ESR? query and verify that the error bit returned and, if set, check the event queue to ascertain the reason for the error. If the error was caused by an interrupted query, then the asynchronous data transfer had not completed when the *ESR? query was sent. In this case, you might need increase your program's time-out value to ensure that all data is transferred and read.*

CURVENext? (Query Only)

This query-only command returns unique waveform data from the instrument. This query performs just like CURVe?, except multiple uses guarantee that the waveform returned is always a new acquisition since the previous CURVENext?. Note that if the instrument is acquiring waveform records at a slow rate (high resolution mode), you must configure the controller for long timeout thresholds. Data will not be transferred until a new waveform is acquired since the previous :CURVENext? response.

NOTE. *This command does not work with Single Sequence.*

Group Waveform Transfer

Syntax `CURVENext?`

Related Commands CURVe?, DATa:DESTination, DATa:SOUrce, DATa:STARt, DATa:STOP, WFMOutpre?

Examples `CURVENEXT?` This query with ASCII encoding, start and stop of 1 and 10 respectively, and a width set to 1 might return `:CURVENEXT 61,62,61,60,60,-59,-59,-58,-58,-59`.

CURVEStream

This query continuously transfers waveform data from the instrument as it is acquired. This command puts the instrument into a talk-only mode, allowing the controller to receive waveform records as fast as (and as soon as) they are

acquired. Use the DATa:SOUrce command to specify the waveform sources. The command does the same thing as the CURVe command.

- Control of the instrument through the user interface or other external client is not possible while in streaming mode.
- The GPIB controller must take the instrument out of this continuous talking mode to terminate the query and allow other input sources to resume communication with the instrument. The following options are available to transition out of streaming curve mode: send a device clear over the bus or send another query to the instrument (a MEPE Query Interrupted error will occur, but the instrument will be placed back into its normal talk/listen mode).
- Turning the waveform screen display mode off (:DISplay:WAVEform OFF) will increase waveform throughput during streaming mode.
- While in streaming mode, two extreme conditions can occur. If the waveform records are being acquired slowly (high resolution), configure the controller for long time-out thresholds, as the data is not sent out until each complete record is acquired. If the waveform records are being acquired rapidly (low resolution), and the controller is not reading the data off the bus fast enough, the trigger rate is slowed to allow each waveform to be sent sequentially.

NOTE. *This command does not work with Single Sequence.*

Group Waveform Transfer

Syntax

```
CURVEStream {<Block>|<asc curve>}
CURVEStream?
```

Related Commands CURVe, DATa:DESTination, DATa:SOUrce, DATa:STARt, DATa:STOP, SAVe:WAVEform, SAVe:WAVEform:FILEFormat, WFMInpre?, WFMInpre:BYT_Nr, WFMOutpre?, WFMInpre:NR_FR?, WFMOutpre:NR_FR?

Arguments See the CURVe command.

Returns Waveform data in its native binary format. To save processing time, DATa:ENCdg FASTEST is set and waveform data is shipped in its raw native format.

DATa

This command sets or queries the format and location of the waveform data that is transferred with the CURVe command.

Group Waveform Transfer

Syntax
```
DATa {INIT|SNAp}
DATa?
```

Related Commands CURVe, DATa:STARt, DATa:STOP, DATa:ENCdg, SAVe:WAVEform:FILEFormat, WFMInpre:NR_FR?, WFMInpre:NR_Pt, WFMOutpre:NR_FR?, WFMOutpre:NR_Pt?

Arguments `INIT` initializes the waveform data parameters to their factory defaults.

`SNAp` sets DATa:STARt and DATa:STOP to match the current V Bar/Screen/Waveform cursor positions.

Examples `DATA INIT` initializes the waveform data parameters to their factory defaults.

`DATA?` might return `:DATA:DESTINATION REF1;ENCDG RIBINARY;SOURCE CH1;START 1;STOP 1000;FRAMESTART 1;FRAMESTOP 1000`.

DATa:DESTination

This command sets or queries the reference memory location for storing waveform data that is transferred into the instrument by the CURVe command.

Group Waveform Transfer

Syntax
```
DATa:DESTination REF<x>
DATa:DESTination?
```

Related Commands CURVe

Arguments `REF<x>` is the reference where the waveform will be stored. The reference number is specified by x, which ranges from 1 through 4.

Examples `DATA:DESTINATION?` might return `:DATA:DESTINATION REF3`, indicating that reference 3 is the currently selected reference memory location for waveform data. `DATA:DESTINATION REF1` indicates that incoming waveform data be stored in reference 1.

DATa:ENCdg

This command sets or queries the format of outgoing waveform data. This command is equivalent to setting WFMOutpre:ENCdg, WFMOutpre:BN_Fmt, and WFMOutpre:BYT_Or. Setting the DATa:ENGdg value causes the corresponding WFMOutpre values to be updated and vice versa.

NOTE. *Values are constrained (for outbound data) to the format of the data specified by DATa:SOUrce.*

Group Waveform Transfer

Syntax

```
DATa:ENCdg
{ASCIi|FAStest|RIBinary|RPBinary|FPBinary|SRIbinary
|SRPbinary|SFPbinary}
DATa:ENCdg?
```

Related Commands WFMOutpre:ENCdg, WFMOutpre:BN_Fmt, WFMOutpre:BYT_Or

Arguments

- `ASCIi` specifies the ASCII representation of signed INT, FLOAT. 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 first waveform specified in the DATA:SOUrce list.

  :ENCdg will always be BIN, :BYT_Or will always be LSB, but :BN_Fmt and :BYT_Nr will depend on the first DATa:SOUrce waveform. :BN_Fmt will be RI unless the waveform is internally stored as a floating point number, in which case the FP format will be used.
- `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. When :BYT_Nr is 8, then the waveform being queried is set to Fast Acquisition mode. 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. When :BYT_Nr is 8, then the waveform being queried is set to Fast Acquisition mode. 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.

- FPBinary specifies the floating point (width = 4) data.

  The range is from -3.4×10^{38} to 3.4×10^{38}. The center of the screen is 0. The upper limit is the top of the screen and the lower limit is the bottom of the screen.

  The FPBinary argument is only applicable to math waveforms or ref waveforms saved from math waveforms.

- 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.

- SFPbinary specifies floating point data in IBM PC format. The SFPbinary argument only works on math waveforms or ref waveforms saved from math waveforms.

Table 2-44: DATa and WFMOutpre Parameter Settings

	WFMOutpre Settings			
DATa:ENCdg Setting	**:ENCdg**	**:BN_Fmt**	**:BYT_Or**	**:BYT_NR**
ASCii	ASC	N/A	N/A	1,2,4,8
FAStest	BIN	RI/FP	MSB	1,2,4
RIBinary	BIN	RI	MSB	1,2,8
RPBinary	BIN	RP	MSB	1,2,8
FPBinary	BIN	FP	MSB	4
SRIbinary	BIN	RI	LSB	1,2,8
SRPbinary	BIN	RP	LSB	1,2,8
SFPbinary	BIN	FP	LSB	4

Examples

DATA:ENCDG RPBinary sets the data encoding format to be a positive integer where the most significant byte is transferred first.

DATA:ENCDG? might return :DATa:ENCDG SRPBINARY for the format of the outgoing waveform data.

DATa:FRAMESTARt

This command sets or queries the starting data frame for waveform transfers. This command allows for transferring a subset of frames of a FastFrame waveform from the instrument (CURVe? query). Any value set by this command is ignored when transferring waveform data to the instrument (CURVe command).

Group Waveform Transfer

Syntax

```
DATa:FRAMESTARt <NR1>
DATa:FRAMESTARt?
```

Arguments <NR1> is the first data frame that is transferred, which ranges from 1 to the number of acquired frames. Data is transferred from <NR1> to DATa:FRAMESTOP or the number a acquired frames, whichever is less. If <NR1> is greater than the number of acquired frames, then you will receive the last acquired frame. If no frames were acquired, then you receive the first frame which contains all nulls.

Returns The starting data frame for waveform transfer.

Examples DATa:FRAMESTARt 10 specifies that the waveform transfer will begin with frame 10.

DATa:FRAMESTARt? might return :DATA:FRAMESTART 214 indicating the first waveform frame that will be transferred.

DATa:FRAMESTOP

This command sets or queries the ending data frame for waveform transfers. This command allows for transferring a subset of frames of a FastFrame waveform from the instrument (CURVe? query). Any value set by this command is ignored when transferring waveform data to the instrument (CURVe command).

Group Waveform Transfer

Syntax

```
DATa:FRAMESTOP <NR1>
DATa:FRAMESTOP?
```

Arguments <NR1> is the last data frame that is transferred, which ranges from 1 to the number of acquired frames. If <NR1> is greater than the number of acquired frames, then you will receive up to the number of acquired frames. If both

DATA:FRAMESTART and DATA:FRAMESTOP are greater than the number of acquired frames, then you will receive the last acquired frame. If no frames were acquired, then you receive the first frame which contains all nulls.

DATA:FRAMESTART and DATA:FRAMESTOP are order independent. The smaller number is used as in CURVe and SAVe:WAVEform queries.

To always transfer complete waveforms, set DATA:FRAMESTART to 1 and DATA:FRAMESTOP to the maximum number of acquired frames, or larger.

Returns The last data frame that will be transferred.

Examples DATa:FRAMESTOP 1500 specifies that the waveform transfer will stop with frame 1500.

DATa:FRAMESTOP? might return :DATA:FRAMESTOP 14900 indicating the last waveform frame that will be transferred.

DATa:SOUrce

This command sets or queries the location of waveform data that is transferred from the instrument by the CURVe? Query.

Group Waveform Transfer

Syntax

```
DATa:SOUrce <wfm>[<,><wfm>]
DATa:SOUrce?
```

Related Commands CURVe, DATa

Arguments <wfm> is the location of the waveform data that will be transferred from the instrument to the controller. It can consist of CH<x>, MATH<x>, REF<x>, DIGITALALL. Note that digital data is transferred as 16-bit data, with the least-significant bit representing D0, and the most-significant bit representing D15.

<wfm> can consist of the following:

CH<x> selects the specified analog channel as the source.

MATH<x> selects the specified reference waveform as the source.

REF<x> selects the specified reference waveform as the source.

DIGITALALL selects digital waveforms as the source. The Digital data is transferred as 16-bit data, with the least-significant bit representing D0, and the

most-significant bit representing D15. The LSB always contains D0-D7 and MSB always contains D8-D15 data.

Examples

`DATA:SOURCE CH1` specifies that the CH1 waveforms will be transferred in the next CURVe? query.

`DATA:SOURCE?` might return `:DATA:SOURCE REF3`, indicating that the source for the waveform data which is transferred using a CURVe? query is reference 3.

DATa:STARt

This command sets or queries the starting data point for waveform transfer. This command allows for the transfer of partial waveforms to and from the instrument.

Group

Waveform Transfer

Syntax

```
DATa:STARt <NR1>
DATa:STARt?
```

Related Commands

CURVe, DATa, DATa:STOP, WFMInpre:NR_Pt, WFMOutpre:NR_Pt?, WFMInpre:NR_FR?, WFMOutpre:NR_FR?

Arguments

`<NR1>` is the first data point that will be transferred, which ranges from 1 to the record length. Data will be transferred from `<NR1>` to `DATa:STOP` or the record length, whichever is less. If `<NR1>` is greater than the record length, the last data point in the record is transferred.

`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.

Examples

`DATA:START 10` specifies that the waveform transfer will begin with data point 10.

`DATA:START?` might return `:DATA:START 214`, indicating that data point 214 is the first waveform data point that will be transferred.

DATa:STOP

This command sets or queries the last data point that will be transferred when using the CURVe? query. This command allows for the transfer of partial waveforms to the controller.

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.

NOTE. *When using the CURVe command, DATa:STOP is ignored and WFMInpre:NR_Pt is used.*

Group Waveform Transfer

Syntax
```
DATa:STOP <NR1>
DATa:STOP?
```

Related Commands CURVe, DATa, DATa:STARt, WFMInpre:NR_Pt, WFMOutpre:NR_Pt?, WFMInpre:NR_FR?, WFMOutpre:NR_FR?

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 15000 specifies that the waveform transfer will stop at data point 15000.

DATA:STOP? might return :DATA:STOP 14900, indicating that 14900 is the last waveform data point that will be transferred.

DATa:SYNCSOUrces

This command sets or queries if the data sync sources are on or off.

Group Waveform Transfer

Syntax
```
DATa:SYNCSOUrces {<NR1>|OFF|ON}
DATa:SYNCSOUrces?
```

Arguments

NR1 = 0 specifies that the data sync sources are not available; any other value specifies that the data sync sources are available.

OFF specifies that the data sync sources are not available.

ON specifies that the data sync sources are available.

Examples

DATA:SYNCSOURCES ON sets the data sync sources are available.

DATA:SYNCSOURCES? might return DATA:SYNCSOURCES OFF, indicating that the data sync sources are not available.

DATE

This command sets or queries the date that the instrument can display. This command is equivalent to selecting Set Date & Time from the Utilities menu and then setting the fields in the Date group box.

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"2000-01-24" specifies that the date is set to January 24, 2000.

DATE? might return :DATE 2000-01-24, indicating the current date is set to January 24, 2000.

*DDT

This command allows you to specify a command or a list of commands that are executed when the instrument receives a *TRG command or the GET GPIB 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.

DELEte:SETUp (No Query Form)

This command (no query form) changes the setup to reference the factory setup instead of the specific user setup slot. The content of the setup slot is unchanged, but the data will no longer be accessible to you.

This command is equivalent to selecting Delete from the File menu, and then clicking the specific setup you want to delete (user setups are shown as Setup-User and are ordered in the list from 1 through 10, if defined) or All Setups.

NOTE. *The setup information cannot be recovered once it is deleted.*

Group Save and Recall

Syntax

```
DELEte:SETUp {ALL|<NR1>}
```

Related Commands *RCL, RECAll:SETUp, *RST, *SAV, SAVe:SETUp, *SDS

Arguments

ALL deletes all the stored setups.

<NR1> specifies a setup storage location to delete. Setup storage location values range from 1 through 10; using an out-of-range value causes an error.

Examples `DELETE:SETUP ALL` deletes all stored setups. All ten storage locations are initialized to the factory default setup.

DELEte:WAVEform (No Query Form)

This command (no query form) deletes one or all stored reference waveforms from memory.

This command is equivalent to selecting Delete from the File menu, and then choosing the reference waveform you want to delete; choosing All Refs deletes all of the reference waveforms.

Group Save and Recall

Syntax
```
DELEte:WAVEform {ALL|REF<x>}
```

Related Commands RECAll:WAVEform, SAVe:WAVEform

Arguments `ALL` specifies to delete all the stored reference waveforms.

`REF<x>` specifies to delete one of the reference memory locations. Reference memory location values range from 1 through 4.

Examples `DELETE:WAVEFORM ALL` removes all waveforms stored in reference memory.

`DELETE:WAVEFORM REF2` removes the waveform stored at REF2.

DESE

This command sets and queries 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 more 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> The binary bits of the DESER are set 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:CONTROL:HALT

This command determines or queries whether the next execution of diagnostics looping will stop on the first diagnostic failure that occurs or will continue to loop on the selected set of diagnostic functions. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then enabling Halt on Fail.

Group

Diagnostics

Syntax

```
DIAg:CONTROL:HALT {<NR1>|ON|OFF}
DIAg:CONTROL:HALT?
```

Related Commands

DIAg:CONTROL:LOOP

Arguments

<NR1> = 1 enables the halt function; any other value disables the halt function.

ON enables the halt function, causing the execution of diagnostics looping to halt at the first diagnostic failure that occurs.

OFF disables the halt looping function, allowing the instrument to continue to loop on the entire set of diagnostics, even if diagnostic failure occurs.

Examples

DIAG:CONTROL:HALT ON enables the halt function, causing the execution of diagnostics looping to halt at the first diagnostic failure that occurs.

DIAG:CONTROL:HALT? might return :DIAG:CONTROL:HALT 0, indicating that the halt function is disabled.

DIAg:CONTROL:LOOP

This command determines or queries whether the next execution of diagnostics executes once or continuously loops on the selected set of diagnostics (assuming the halt control is set to off using the DIAg:CONTROL:HALT command or that the halt control is set to ON but no failures occur). This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then enabling Loop Control.

Group Diagnostics

Syntax

```
DIAg:CONTROL:LOOP {<NR1>|ON|OFF}
DIAg:CONTROL:LOOP?
```

Related Commands DIAg:CONTROL:HALT, DIAg:STOP

Arguments <NR1> = 1 enables the loop function; any other value disables the loop function.

ON enables the loop function, causing the execution of diagnostics to continuously loop.

OFF disables the loop function, causing the instrument to execute the entire set of diagnostics once and then halt.

Examples DIAG:CONTROL:LOOP ON enables the loop function.

DIAG:CONTROL:LOOP? might return :DIAG:CONTROL:LOOP 0, indicating that looping on diagnostics is disabled.

DIAg:EXECUTE (No Query Form)

This command (no query form) starts the execution of the currently selected set of diagnostics. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then pressing Run.

Group Diagnostics

Syntax

```
DIAg:EXECUTE
```

Related Commands DIAg:STATE

Examples DIAG:EXECUTE starts the execution of the entire set of diagnostics.

DIAg:FAILURES:CLEAR

This command sets and queries the clearing of pass/fail information from data structures, not the Event Log, at the start of diagnostic tests.

Group Diagnostics

Syntax
```
DIAg:FAILURES:CLEAR {<NR1>|OFF|ON}
DIAg:FAILURES:CLEAR?
```

Arguments <NR1> = 0 turns off the clearing the pass/fail information at the start of tests; any other value turns on the clearing of pass/fail information.

OFF does not clear pass/fail information at the start of tests.

ON clears pass/fail information from data structures, not the Event Log, at the start of tests.

Examples DIAG:FAILURES:CLEAR ON clears pass/fail information on the start of tests.

DIAG:FAILURES:CLEAR? might return DIAG:FAILURES:CLEAR1 indicating that pass/fail information will be cleared at the start of tests.

DIAg:ITEM? (Query Only)

This query-only command returns the diagnostics settings. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu, and then reviewing the diagnostics settings.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax
```
DIAg:ITEM? <NR1>
```

Related Commands DIAg:ITEM:FAILURES?, DIAg:ITEM:NAMe?, DIAg:ITEM:RESULT?, DIAg:ITEM:SUBITEMS?

Arguments <NR1> sets the index item about which data will be returned, which ranges from 0 through 15.

Examples DIAG:ITEM? 2 might return :DIAG:ITEM 2,"Acquisition","Pass",0,8, indicating that the acquisition test passed and that there were 0 failures of the 8 subitems for that test.

DIAg:ITEM:FAILURES? (Query Only)

This query-only command returns the total number of failures. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax DIAg:ITEM:FAILURES? <NR1>

Related Commands DIAg:ITEM?, DIAg:ITEM:NAMe?, DIAg:ITEM:RESULT?, DIAg:ITEM:SUBITEMS?

Arguments <NR1> sets the index item about which data will be returned, which ranges from 0 through 15.

Examples DIAG:ITEM:FAILURES? 2 might return :DIAG:ITEM:FAILURES 2,0, indicating that there were 0 failures for item 2.

DIAg:ITEM:NAMe? (Query Only)

This query-only command returns the name of the selected menu item. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Subsystem, Area and Test settings.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax DIAg:ITEM:NAMe? <NR1>

Related Commands DIAg:ITEM?, DIAg:ITEM:FAILURES?, DIAg:ITEM:RESULT?, DIAg:ITEM:SUBITEMS?

Arguments <NR1> sets the index item about which data will be returned, which ranges from 0 through 15.

Examples DIAG:ITEM:NAME? 2 might return :DIAG:ITEM:NAME 2,"Acquisition" indicating that the selected menu item (2) is Acquisition.

DIAg:ITEM:RESULT? (Query Only)

This query-only command returns the result from the last execution of the item. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax DIAg:ITEM:RESULT? <NR1>

Related Commands DIAg:ITEM?, DIAg:ITEM:NAMe?, DIAg:ITEM:FAILURES?, DIAg:ITEM:SUBITEMS?

Arguments <NR1> sets the index item about which data will be returned, which ranges from 0 through 15.

Examples DIAG:ITEM:RESULT? 2 might return :DIAG:ITEM:RESULT 2,"Pass", indicating that the item passed during the last execution.

DIAG:ITEM:RESULT? 2 might return :DIAG:ITEM:RESULT 2,"Fail", indicating that the item failed during the last execution.

DIAG:ITEM:RESULT? 2 might return :DIAG:ITEM:RESULT 2," ", indicating that the item was not run.

DIAg:ITEM:SUBITEMS? (Query Only)

This query-only command returns the number of subitems associated with the item. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu, choosing the Subsystem, Area or Test setting and then reviewing the resulting subitems.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax

```
DIAg:ITEM:SUBITEMS? <NR1>
```

Related Commands DIAg:ITEM?, DIAg:ITEM:NAMe?, DIAg:ITEM:FAILURES?, DIAg:ITEM:RESULT?

Arguments <NR1> sets the index item about which data will be returned, which ranges from 0 through 15.

Examples DIAG:ITEM:SUBITEMS? 2 might return :DIAG:ITEM:SUBITEMS 2,8, indicating that there are 8 subitems associated with the selected item.

DIAg:LEVEL

This command sets or queries the selected level of diagnostic test hierarchy. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

Group Diagnostics

Syntax

```
DIAg:LEVEL {AREA|SUBSYS|TEST}
DIAg:LEVEL?
```

Arguments AREA sets diagnostic testing to the area level.

SUBSYS sets diagnostic testing to the subsystem level.

TEST sets diagnostic testing to the test level.

Examples DIAG:LEVEL AREA sets the level of diagnostic test hierarchy to Area.

DIAG:LEVEL? might return :DIAG:LEVEL SUBSYSI, indicating that the current level of diagnostic test hierarchy is Subsys.

DIAg:LOOPS? (Query Only)

This query-only command returns the number of times that the selected diagnostics set was completed during the last diagnostic execution. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Elapsed Loops.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax `DIAg:LOOPS?`

Examples `DIAG:LOOPS?` might return `:DIAG:LOOPS 5` indicating that the selected set of diagnostics was completed five times during the last diagnostic execution.

DIAg:NAMe? (Query Only)

This query-only command returns the names of the subsystem, area, and test of the current diagnostic test. This command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax `DIAg:NAMe?`

Related Commands DIAg:NAMe:AREA?, DIAg:NAMe:SUBSYS?, DIAg:NAMe:TEST?

Examples `DIAG:NAME?` might return `:DIAG:NAME:SUBSYS "Processor";AREA "Memory";TEST"DRAMWalk1"`, indicating the subsystem name, area name, and test name of the currently selected diagnostic test.

DIAg:NAMe:AREA? (Query Only)

This query-only command returns the selected area of the current diagnostic test. There are three levels of diagnostic test hierarchy: subsystem, area and test. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax `DIAg:NAMe:AREA?`

Related Commands DIAg:NAMe?, DIAg:NAMe:SUBSYS?, DIAg:NAMe:TEST?

Examples `DIAG:NAME:AREA?` might return `:DIAG:NAME:AREA "Memory"` indicating the selected area name of the current diagnostic test.

DIAg:NAMe:SUBSYS? (Query Only)

This query-only command returns the subsystem of the current diagnostic test. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax `DIAg:NAMe:SUBSYS?`

Related Commands DIAg:NAMe?, DIAg:NAMe:AREA?, DIAg:NAMe:TEST?

Examples `DIAG:NAME:SUBSYS?` might return `:DIAG:NAME:SUBSYS "Acquisition"`, indicating the subsystem name of the current diagnostic test.

DIAg:NAMe:TEST? (Query Only)

This query-only command returns the name of the current diagnostic test. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax `DIAg:NAMe:TEST?`

Related Commands DIAg:NAMe?, DIAg:NAMe:AREA?, DIAg:NAMe:SUBSYS?

Examples `DIAG:NAME:TEST?` might return `:DIAG:NAME:TEST "diagDataFormatConf"`, indicating the test name of the current diagnostic test.

DIAg:NUMITEMS? (Query Only)

This query-only command returns the number of items on the currently selected level of test hierarchy, which ranges from 1 through 15. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax `DIAg:NUMITEMS?`

Examples `DIAG:NUMITEMS?` might return `:DIAG:NUMITEMS 7`, indicating the number of items on the currently selected level of test hierarchy.

DIAg:RESults? (Query Only)

This query-only command returns an abbreviated status about the results of the last diagnostic (or test) execution. For a more explanatory status message, use the `DIAg:RESults:VERBose?` query. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax `DIAg:RESults?`

Related Commands DIAg:RESults:VERBose?

Examples `DIAG:RESULTS?` might return `:DIAG:RESULTS "135"`, indicating the specific failure code of the first test failure that occurred.

`DIAG:RESULTS?` might return `:DIAG:RESULTS "Pass"`, indicating that the last diagnostic test passed.

DIAG:RESULTS? might return :DIAG:RESULTS "*****", indicating that the diagnostic test was not run.

DIAg:RESults:VERBose? (Query Only)

This query-only command returns a more explanatory message about the results of the last diagnostic (or test) execution than the DIAg:RESults? query. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

This query-only command can be issued while diagnostics are still in progress.

Group Diagnostics

Syntax DIAg:RESults:VERBose?

Related Commands DIAg:RESults?

Examples DIAG:RESULTS:VERBOSE? might return

:DIAG:RESULTS:VERBOSE "DIAG ABORTED, ERROR 15 in A2D-Dmux 1 interconnects".

DIAg:SELect:ALL (No Query Form)

This command (no query form) selects all available diagnostics. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then choosing ALL from the Subsystem, Area and Test pull-down lists.

Group Diagnostics

Syntax DIAg:SELect:ALL ALL

Related Commands DIAg:NAMe:AREA?, DIAg:NAMe:SUBSYS?, DIAg:NAMe:TEST?, DIAg:EXECUTE

Arguments ALL selects all available diagnostics.

Examples DIAG:SELECT:ALL ALL selects all available diagnostics.

DIAg:SELect:AREA

This command selects or queries an available diagnostic area. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then selecting an Area from the pull-down list.

NOTE. *This command should be executed only if* `DIAg:LEVEL` *is currently set to AREA.*

Group Diagnostics

Syntax

```
DIAg:SELect:AREA <NR1>
DIAg:SELect:AREA?
```

Related Commands DIAg:SELect:SUBSYS, DIAg:SELect:TEST, DIAg:SELect:LAST

Arguments `<NR1>` selects a diagnostic area by number, which can range from 0 (zero selects all) through 15 (as specified by DIAg:NUMITEMS?).

Examples `DIAG:SELECT:AREA 1` selects Area 1 as the area to be diagnosed.

`DIAG:SELECT:AREA?` might return `:DIAG:SELECT:AREA 0`, indicating that all areas are selected.

DIAg:SELect:LAST

This command selects one or more diagnostic menu items to be executed via the DIAg:EXECUTE command. If you specify `DIAg:LEVEL SUBSYS`, then menu items come from this diagnostic level and are limited to the value returned by the `DIAg:NUMITEMS?` query. For example, if the return from the `DIAg:NUMITEMS?` query is `:DIAG:NUMITEMS 5`, specifying `DIAg:SELECT:SUBSYS 2` indicates that diagnostics will start from subsystem 2 and that you can specify a range from 2 through 5 for `DIAg:SELect:LAST`.

If you enter `:DIAg:SELect:LAST 2`, only subsystem 2 will be executed. `DIAg:SELect:LAST 4`, subsystems 2 *through* 4 will be executed.

Group Diagnostics

Syntax

```
DIAg:SELect:LAST <NR1>
DIAg:SELect:LAST?
```

Related Commands DIAg:EXECUTE, DIAg:LEVEL, DIAg:NAMe:AREA?, DIAg:NAMe:SUBSYS?, DIAg:NAMe:TEST?, DIAg:NUMITEMS?, DIAg:SELect:SUBSYS

Arguments <NR1> selects an integer that identifies the number of the last item that will be executed when the DIAg:EXECUTE command is run.

Examples DIAG:SELECT:LAST 2 specifies that (based on the example in the description above) only subsystem 2 will be executed.

DIAG:SELECT:LAST? might return :DIAG:SELECT:LAST 5, indicating that 5 is the last item of a group of items from the current level of test hierarchy.

DIAg:SELect:SUBSYS

This command selects or queries the available subsystem. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then choosing a Subsystem from the drop-down list.

NOTE. *This command should be executed only if* ***DIAG:LEVEL*** *is currently set to SUBSYS.*

Group Diagnostics

Syntax
```
DIAg:SELect:SUBSYS <NR1>
DIAg:SELect:SUBSYS?
```

Related Commands DIAg:SELect:AREA, DIAg:SELect:LAST

Arguments <NR1> selects a subsystem by number, which can range from 0 (zero selects ALL) through 15 (as limited by the return from DIAg:NUMITEMS?).

Examples DIAG:SELECT:SUBSYS 1 selects Subsystem 1 as the subsystem to be diagnosed.

DIAG:SELECT:SUBSYS? might return :DIAG:SELECT:SUBSYS 0, indicating that all subsystems are selected.

DIAg:SELect:TEST

This command selects or queries one of the available tests. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then choosing a Test from the drop-down list.

NOTE. *This command should be executed only if* `DIAg:LEVEL` *is currently set to TEST.*

Group Diagnostics

Syntax
```
DIAg:SELect:TEST <NR1>
DIAg:SELect:TEST?
```

Related Commands DIAg:SELect:AREA, DIAg:SELect:LAST, DIAg:SELect:SUBSYS

Arguments `<NR1>` selects a test by number, which can range from 0 (zero selects ALL) through 15 (as limited by the return from DIAg:NUMITEMS?).

Examples `DIAG:SELECT:TEST 1` selects Test 1 as the test to be executed.

`DIAG:SELECT:TEST?` might return `:DIAG:SELECT:TEST 0`, indicating all tests have been selected.

DIAg:STATE

This command changes the instrument operating state. Depending on the argument, diagnostics capabilities are either turned on or off. This command is equivalent to opening the DIAg:STATE dialog (ON) or closing it (OFF).

Group Diagnostics

Syntax
```
DIAg:STATE {<NR1>|EXECUTE|ON|OFF}
DIAg:STATE?
```

Related Commands TEST:STOP

Arguments `<NR1>` = 0 disables diagnostics capabilities and returns the instrument to a normal operating state; any other value enables diagnostics.

`EXECUTE` starts execution of the diagnostics.

`ON` puts the instrument into the state in which diagnostics can be run. This argument is thrown automatically if the DIAg:EXECUTE command is executed.

`OFF` disables diagnostics capabilities and returns the instrument to a normal operating state.

Examples `DIAG:STATE OFF` turns off diagnostics capabilities and returns the instrument to a normal operating state.

`DIAG:STATE?` might return `:DIAG:STATE 0`, indicating that diagnostics are disabled.

DIAg:STOP (No Query Form)

This command (no query form) causes diagnostics (or test) execution to terminate at the end of the next low-level test. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then clicking Abort.

Group Diagnostics

Syntax `DIAg:STOP`

Related Commands TEST:STOP

Examples `DIAG:STOP` terminates diagnostics (or test) execution at the end of the next low-level test.

DISplay? (Query Only)

This query-only command returns the current Display settings. This command is equivalent to selecting Display Setup from the Display menu and then viewing the contents of each tab.

Group Display Control

Syntax `DISplay?`

Examples `DISPLAY?` might return `:DISPLAY:CLOCK 1;COLOR:PALETTE:IMAGEVIEW MONOGREEN; RECORDVIEW TEMPERATURE;USER:GRATICULE 165,50,15;CH1 180,50,100; CH2 300,50,100;CH3 60,50,100;CH4 240,50,100;REF1 0,90,0; REF2 0,90,100;REF3 60,90,100;REF4 240,90,100;MATH1 160,50,100; MATH2 40,60,100;MATH3 120,60,100;MATH4 195,50,100; HISTOGRAM 320,50,100;CARET 150,50,100;MASK 0,25,75;MASKHIGHLIGHT 140,50,100;:DISPLAY:COLOR:MATHCOLOR DEFAULT;REFCOLOR DEFAULT; :DISPLAY:FILTER SINX;FORMAT`

```
YT;GRATICULE IRE;INTENSITY :WAVEFORM:IMAGEVIEW
81.0000;RECORDVIEW 81.0000;:DISPLAY :INTENSITY:AUTOBRIGHT
0; :DISPLAY:PERSISTENCE OFF;STYLE DOTS;TRIGBAR OFF;TRIGT
1; CURSORTICK LONG;VARPERSIST 2.6000;SCREENTEXT:STATE
1; LABEL1:NAME "";XPOS 100;YPOS 5;:DISPLAY:SCREENTEXT
:LABEL2:NAME "THIS IS SCREEN TEXT";XPOS 100;YPOS 20;
:DISPLAY:SCREENTEXT:LABEL3:NAME "";XPOS 100;YPOS 35;
:DISPLAY:SCREENTEXT:LABEL4:NAME "";XPOS 100;YPOS 50;
:DISPLAY:SCREENTEXT:LABEL5:NAME "";XPOS 100;YPOS 343;
:DISPLAY:SCREENTEXT:LABEL6:NAME "";XPOS 100;YPOS 80;
:DISPLAY:SCREENTEXT:LABEL7:NAME "";XPOS 100;YPOS 95;
:DISPLAY:SCREENTEXT:LABEL8:NAME "";XPOS 100;YPOS 110;
:DISPLAY:WAVEFORM 1.
```

DISplay:CLOCk

This command sets or queries the display of the date and time. This is equivalent to selecting Display Date & Time from the Display menu. The query form of this command returns an ON (1) or an OFF (0).

Group Display Control

Syntax

```
DISplay:CLOCk {<NR1>|ON|OFF}
DISplay:CLOCk?
```

Related Commands DATE, TIME

Arguments

<NR1> = 0 disables display of date and time; any other value enables display of date and time.

ON enables display of date and time.

OFF disables 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:COLOr? (Query Only)

This query-only command returns the settings from the PALETTE, MATHCOLOR and REFCOLOR commands. This is equivalent to selecting Colors from the Display menu.

Group Display Control

Syntax

```
DISplay:COLOr?
```

Arguments None

Examples `DISPLAY:COLOR?` might return `:DISPLAY:COLOR:PALETTE:IMAGEVIEW TEMPERATURE;RECORDVIEW NORMAL;USER:GRATICULE 165`,50,15;CH1 180,50,100;CH2 300,50,100;CH3 60,50,100;CH4 240,50,100;REF1 0,90,0;REF2 0,90,100;REF3 60,90,100;REF4 240,90,100;MATH1 160,50,100;MATH2 40,60,100;MATH3 120,60,100;MATH4 195,50,100;HISTOGRAM 320,50,100;CARET 150,50,100;MASK 0,25,75;MASKHIGHLIGHT 140,50,100;:DISPLAY:COLOR:MATHCOLOR DEFAULT;REFCOLOR DEFAULT indicating that the primary source waveform color is used for each math trace.

DISplay:COLOr:MATHCOLOr

This command sets or queries the color to be used for math traces, either in the standard palette's nominal Math color, or according to the color of the source waveform. This command is equivalent to selecting Display Setup from the Display menu and then choosing the Colors tab.

Group Display Control

Syntax

```
DISplay:COLOr:MATHCOLOr {DEFAULT|INHERIT}
DISplay:COLOr:MATHCOLOr?
```

Related Commands DISplay:COLOr:PALEtte:IMAGEView, DISplay:COLOr:PALEtte:RECORDView

Arguments `DEFAULT` sets color math traces in nominal palette math color, which is red.

`INHERIT` sets color math traces in the source waveform color. Math waveforms are drawn in the same color as their primary source waveform.

Examples

DISPLAY:COLOR:MATHCOLORDEFAULT sets the color math traces in the nominal palette math color.

DISPLAY:COLOR:MATHCOLOR? might return :DISPLAY:COLOR:MATHCOLOR INHERIT, indicating that the primary source waveform color is used for each math trace.

DISplay:COLOr:PALEtte:IMAGEView

This command sets or queries the color palette for all image view (DPO and WfmDB) traces.

Group

Display Control

Syntax

```
DISplay:COLOr:PALEtte:IMAGEView
{NORMal|MONOGREEN|MONOGRAY|TEMPErature|SPECTral|USEr}
DISplay:COLOr:PALEtte:IMAGEView?
```

Arguments

NORMal colors traces according to their channel. This is the default color palette.

MONOGREEN colors traces green, emulating a traditional instrument color palette.

MONOGRAY colors traces gray, emulating a monochrome instrument.

TEMPErature

All traces share a multicolored palette, where “intensity” is represented by hue; blue for least frequently hit, red for most frequently hit.

SPECTral.

All traces share a multicolored palette, where “intensity” is represented by hue; red for least frequently hit, blue for most frequently hit.

USEr.

All traces share a user-defined palette.

Examples

DISPLAY:COLOR:PALETTE:IMAGEVIEWMONOGRAY sets the palette for all display types.

DISPLAY:COLOR:PALETTE:IMAGEVIEW? might return :DISPLAY:COLOR:PALETTE:IMAGEVIEW NORMAL, indicating that the Normal color palette is currently selected, which colors traces according to their channel.

DISplay:COLOr:PALEtte:RECORDView

This command sets or queries the color palette for all record view (non image view) waveforms.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:RECORDView
{NORMal|MONOGREEN|MONOGRAY|TEMPErature|SPECTral|USEr}
DISplay:COLOr:PALEtte:RECORDView?
```

Arguments

`NORMal` colors traces according to their channel. This is the default color palette.

`MONOGREEN` colors traces green, emulating a traditional instrument color palette.

`MONOGRAY` colors traces gray, emulating a monochrome instrument.

`TEMPErature` colors all traces using a multicolored palette, where "intensity" is represented by hue; blue for least frequently hit, red for most frequently hit. All traces share this palette.

`SPECTral` colors all traces using a multicolored palette, where "intensity" is represented by hue; red for least frequently hit, blue for most frequently hit. All traces share this palette.

`USEr` colors all traces using a user-defined palette. All traces share this palette.

Examples

`DISPLAY:COLOR:PALETTE:RECORDVIEW MONOGRAY` sets the palette for all display types.

`DISPLAY:COLOR:PALETTE:RECORDVIEW?` might return `:DISPLAY:COLOR:PALETTE:RECORDVIEW NORMAL`, indicating that the Normal color palette is currently selected, which colors traces according to their channel.

DISplay:COLOr:PALEtte:USEr

This command queries the color palette for group settings. It outputs settings from the `DISPlay CARET, CH<x>, GRATICULE, HISTOGRAM, MASK, MASKHIGHLIGHT, MATH<x>,` and `REF<x>` commands.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:USEr RESET
DISplay:COLOr:PALEtte:USEr?
```

Arguments RESET sets all user palettes to their default values.

Examples DISPLAY:COLOR:PALETTE:USER? might return :DISPLAY:COLOR:PALETTE:USER:GRATICULE 165,50,15;CH1 180,50,100;CH2 300 50,100;CH3 60,50,100;CH4 240,50,100;REF1 0,90,0;REF2 0,90,100;REF3 60,90,100;REF4 240,90,100;MATH1 160,50,100;MATH2 40,60,100;MATH3 120,60,100;MATH4 195,50,100;HISTOGRAM 320,50,100;CARET 150,50,100;MASK 0,25,75;MASKHIGHLIGHT 140,50,100.

DISplay:COLOr:PALEtte:USEr:CARet

This command sets or queries the caret color for the user palette. It assigns the HUE, light, Saturation (HLS) triplet used for the caret color. The caret is the solid, inverted delta positioned on the top graticule line, which indicates the trigger position within the waveform record.

Group Display Control

Syntax
```
DISplay:COLOr:PALEtte:USEr:CARet <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:CARet?
```

Arguments <NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples DISPLAY:COLOR:PALETTE:USER:CARET 330,75,100 sets the caret color hue to 330, the lightness to 75, and the saturation to 100.

DISPLAY:COLOR:PALETTE:USER:CARET? might return :DISPLAY:COLOR:PALETTE:USER:CARET 250, 85, 90, indicating that the color hue is set at 250, the lightness is set to 85, and the saturation is set to 90.

DISplay:COLOr:PALEtte:USEr:CH<x>

This command sets or queries the hue-lightness-saturation (HLS) triplet to be used for the specified channel color for the user palette. The channel is specified by x. The value of x can range from 1 through 4.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:USEr:CH<x> <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:CH<x>?
```

Arguments

<NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples

DISPLAY:COLOR:PALETTE:USER:CH1 330,75,100 sets a user palette of hue = 330, lightness = 75, and saturation = 100 for Channel 1.

DISPLAY:COLOR:PALETTE:USER:CH2? might return :DISPLAY:COLOR:PALETTE:USER:CH2 325,65,100 indicating that Channel 2 color for the user palette is set at 325 for hue, 65 for lightness, and 100 for saturation.

DISplay:COLOr:PALEtte:USEr:GRAticule

This command sets or queries the user palette graticule color. It assigns the hue-lightness-saturation triplet to be used for the graticule color for the user palette.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:USEr:GRAticule <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:GRAticule?
```

Arguments

<NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples

DISPLAY:COLOR:PALETTE:USER:GRATICULE 330,75,100 sets the graticule color for the user palette.

DISPLAY:COLOR:PALETTE:USER:GRATICULE? might return :DISPLAY:COLOR:PALETTE:USER:GRATICULE 300,60,85, indicating that the user palette graticule color is assigned a hue of 300, a lightness of 60, and a saturation of 85.

DISplay:COLOr:PALEtte:USEr:HIStogram

This command sets or queries the user palette histogram color. It assigns the hue-saturation-lightness (HLS) triplet to be used for the histogram color for the user palette.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:USEr:HIStogram <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:HIStogram?
```

Arguments

<NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples

`DISPLAY:COLOR:PALETTE:USER:HISTOGRAM 330,75,100` sets the histogram color for the user palette.

`DISPLAY:COLOR:PALETTE:USER:HISTOGRAM?` might return `:DISPLAY:COLOR:PALETTE:USER:HISTOGRAM 330,50,90`, indicating that the user palette histogram color is assigned a hue of 330, a lightness of 50, and a saturation of 90.

DISplay:COLOr:PALEtte:USEr:MASK

This command sets or queries the user palette mask color. It assigns the hue-lightness-saturation (HLS) color for the mask color for the user palette.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:USEr:MASK <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:MASK?
```

Arguments

<NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples

`DISPLAY:COLOR:PALETTE:USER:MASK 330,75,100` sets the mask color for the user palette.

DISPLAY:COLOR:PALETTE:USER:MASK? might return :DISPLAY:COLOR:PALETTE:USER:MASK 330,50,90, indicating that the user palette mask color is assigned a hue of 330, a lightness of 50, and a saturation of 90.

DISplay:COLOr:PALEtte:USEr:MASKHighlight

This command sets or queries the user palette mask hits color. It assigns the hue-lightness-saturation (HLS) triplet to be used for the mask highlight (mask hits) color for the user palette.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:USEr:MASKHighlight <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:MASKHighlight?
```

Arguments

<NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples

DISPLAY:COLOR:PALETTE:USER:MASKHIGHLIGHT 330,75,100 sets the mask highlight (mask hits) color for the user palette.

DISPLAY:COLOR:PALETTE:USER:MASKHIGHLIGHT? might return :DISPLAY:COLOR:PALETTE:USER:MASK:HIGHLIGHT 325,65,100, indicating that the user palette mask highlight color is assigned a hue of 325, a lightness of 65, and a saturation of 100.

DISplay:COLOr:PALEtte:USEr:MATH<x>

This command sets or queries the user palette math colors. It assigns the hue-lightness-saturation (HLS) triplet to be used for the specified math waveform for the user palette. The math waveform is specified by x. The value of x can range from 1 through 4.

Group Display Control

Syntax

```
DISplay:COLOr:PALEtte:USEr:MATH<x> <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:MATH<x>?
```

Arguments <NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples DISPLAY:COLOR:PALETTE:USER:MATH2 330,75,100 sets the color of the specified Math 2 waveform for the user palette.

DISPLAY:COLOR:PALETTE:USER:MATH2? might return :DISPLAY:COLOR:PALETTE:USER:MATH2 345,75,98, indicating that the user palette color of the Math 2 waveform is set to a hue of 345, a lightness of 75, and a saturation of 98.

DISplay:COLOr:PALEtte:USEr:REF<x>

This command sets or queries the user palette reference colors assigned to the reference waveforms. It assigns the hue-lightness-saturation (HLS) triplet to be used for the specified reference waveform color for the user palette. The reference waveform is specified by x. The value of x can range from 1 through 4.

Group Display Control

Syntax
```
DISplay:COLOr:PALEtte:USEr:REF<x> <NR1>,<NR1>,<NR1>
DISplay:COLOr:PALEtte:USEr:REF<x>?
```

Arguments <NR1> Hue. Range of 0 to 360.

<NR1> Lightness. Range of 0 to 100.

<NR1> Saturation. Range of 0 to 100.

Examples DISPLAY:COLOR:PALETTE:USER:REF1 330,75,100 sets the Reference 1 waveform color for the user palette.

DISPLAY:COLOR:PALETTE:USER:REF2? might return :DISPLAY:COLOR:PALETTE:USER:REF2 300,90,100, indicating that the user palette color for Reference 2 waveform is assigned a hue of 300, and lightness of 90, and a saturation of 100.

DISplay:COLOr:PALEtte:USEr:WAVEform

This command sets or queries the user palette waveform colors. It assigns the hue-lightness-saturation (HLS) triplet to be used for the specified waveform for the user palette.

Group Display Control

Syntax
```
DISplay:COLOr:PALEtte:USEr:WAVEform
{HLS|SPECTral|TEMPErature}
DISplay:COLOr:PALEtte:USEr:WAVEform?
```

Arguments `HLS` sets the color of user waveforms to the hue, lightness, and saturation specified by the DISplay:COLOr:PALEtte:USEr:WAVEform commands.

`SPECTral` sets the color of user waveforms to spectral.

`TEMPErature` sets the color of user waveforms to temperature.

Examples `DISPLAY:COLOR:PALETTE:USER:WAVEFORM SPECTRAL` sets the user waveforms color palette to spectral.

`DISPLAY:COLOR:PALETTE:USER:WAVEFORM?` might return `:DISPLAY:COLOR:PALETTE:USER:WAVEFORM TEMPERATURE`, indicating that the color of user waveforms is set to temperature.

DISplay:COLOr:REFCOLOr

This command sets or queries the color to be used for reference traces, either in the standard palette's nominal REF color or according to the color of the source waveform. This command is equivalent to selecting Display Setup from the Display menu and then choosing the Colors tab.

Group Display Control

Syntax
```
DISplay:COLOr:REFCOLOr {DEFAULT|INHERIT}
DISplay:COLOr:REFCOLOr?
```

Related Commands DISplay:INTENSITy:WAVEform:IMAGEView, DISplay:INTENSITy:WAVEform:RECORDView

Arguments `DEFAULT` assigns color reference traces to the nominal palette reference color, which is off-white.

`INHERIT` assigns color reference traces to the source waveform color.

Examples `DISPLAY:COLOR:REFCOLOR DEFAULT` assigns color reference traces to the nominal palette reference color, which is off-white.

DISPLAY:COLOR:REFCOLOR? might return :DISPLAY:COLOR:REFCOLOR INHERIT, indicating that source waveform color is used for reference traces.

DISplay:DESKew

This command controls or queries the state of the Display Only button.

Group Display Control

Syntax

```
DISplay:DESKew {<NR1>|OFF|ON|AUTO}
DISplay:DESKew?
```

Arguments

OFF sets deskew for the acquisition waveform.

ON sets deskew to deskew the display only.

<NR1> = 0 will deskew the acquisition waveform; any other value will deskew the display only.

AUTO automatically selects the deskew setting.

Examples

DISPLAY:DESKEW AUTO automatically selects the deskew setting.

DISPLAY:DESKEW? might return :DISPLAY:DESKEW ON indicating that the deskew is set to deskew the display only.

DISplay:DIGital:HEIght

NOTE. *Not available on some models.*

This command sets or queries the height of the digital input waveform and the label associated with the channel.

Group Digital

Syntax

```
DISplay:DIGital:HEIght {SMAll|LARge|MEDium|XSMAll}
DISplay:DIGital:HEIght?
```

Arguments

SMAll specifies the height of the digital input waveform and the label associated with the channel to small.

LARge specifies the height of the digital input waveform and the label associated with the channel to large.

MEDium specifies the height of the digital input waveform and the label associated with the channel to medium.

XSMAll specifies the height of the digital input waveform and the label associated with the channel to extra small.

Examples

DISPLAY:DIGITAL:HEIGHT MEDIUM sets the height of the digital input waveform and the label associated with the channel to MEDIUM.

DISPLAY:DIGITAL:HEIGHT? might return :DISPLAY:DIGITAL:HEIGHT SMALL, indicating that the height of the digital input waveform and the label associated with the channel is set to SMALL.

DISplay:FILTer

This command sets or queries the type of interpolation to use for the display. Filtering only applies to normal-mode acquisition. The DISplay:FILTer command also provides selection for acquisition interpolation type. This command is equivalent to selecting Waveform Interpolation from the Display menu.

Group

Display Control

Syntax

```
DISplay:FILTer {LINEAr|SINX}
DISplay:FILTer?
```

Arguments

LINEAr specifies linear interpolation, where acquired points are connected with straight lines.

SINX specifies sin(x)/x interpolation, where acquired points are fit to a curve.

Examples

DISPLAY:FILTER SINX specifies sine-curve interpolation, when magnifying waveforms.

DISPLAY:FILTER? might return :DISPLAY:FILTER LINEAR, indicating that straight-line interpolation is specified for magnifying waveforms.

DISplay:FORMat

This command sets or queries the display format. This command is equivalent to selecting Format from the Display menu.

Group Display Control

Syntax

```
DISplay:FORMat {YT|XY|XYZ}
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. The source pairs that make up an XY trace are predefined and are listed in the following table. Selecting one source causes its corresponding source to be implicitly selected, producing a single trace from the two input waveforms.

XYZ argument is available only for four-channel instruments. The argument combines channel 1 and channel 2 for X and Y coordinates and uses channel 3 to provide the intensity value for the sample. XYZ groups channels 1, 2 and 3 to form a single trace. Other channel, math, and reference waveforms are turned off.

Table 2-45: XY Format Pairs

X-Axis Source	Y-Axis Source
CH 1	CH 2
CH 3	CH 4
Ref 1	Ref 2
Ref 3	Ref 4

Examples DISPLAY:FORMAT YT selects a voltage versus time format for the display.

DISPLAY:FORMAT? might return :DISPLAY:FORMAT XY is for the display format, indicating that the display is in the XY mode.

DISplay:GRAticule

This command selects or queries the type of graticule that is displayed. This command is equivalent to selecting Graticule Style from the Display menu.

Group Display Control

Syntax

```
DISplay:GRAticule
{CROSSHair|FRAme|FULl|GRId|IRE|NTSC|MV|PAL}
DISplay:GRAticule?
```

Arguments

`CROSSHair` specifies a frame and cross hairs.

`FRAme` specifies a frame only.

`FULl` specifies a frame, a grid and cross hairs.

`GRId` specifies a frame and grid only.

`IRE` specifies an IRE video graticule, and sets the vertical scale to 143 mV per division.

`NTSC` specifies an NTSC video graticule (same as the `IRE` graticule), and sets the vertical scale to 133 mV per division.

`MV` specifies an mV video graticule and sets the vertical scale to 133 mV per division. This graticule is used to measure PAL standard video signals.

`PAL` specifies a PAL video graticule (same as the mV graticule) and sets the vertical scale to 133 mV per division. This graticule is used to measure PAL standard video signals.

Examples

`DISPLAY:GRATICULE FRAME` sets the graticule type to display the frame only.

`DISPLAY:GRATICULE?` might return `:DISPLAY:GRATICULE FULL`, indicating that all graticule elements are selected.

DISplay:INTENSITy? (Query Only)

This query-only command returns the waveform saturation level and screen saver settings. This command is equivalent to selecting Display Setup from the Display menu and choosing the Appearance tab.

Group

Display Control

Syntax

`DISplay:INTENSITy?`

Related Commands

,DISplay:INTENSITy:WAVEform:IMAGEView, DISplay:INTENSITy:WAVEform:RECORDView

Arguments

None

Examples

`DISPLAY:INTENSITY?` might return `:DISPLAY:INTENSITY:WAVEFORM:IMAGEVIEW 75.0000;RECORDVIEW 10.0000;:DISPLAY:INTENSITY:AUTOBRIGHT 1`.

DISplay:INTENSITy:BACKLight

NOTE. *Not available on some models.*

This command sets or queries the waveform backlight intensity settings.

Group Display Control

Syntax

```
DISplay:INTENSITy:BACKLight {LOW|MEDium|HIGH}
DISplay:INTENSITy:BACKLight?
```

Arguments

`LOW` selects a low brightness level.

`MEDium` selects a moderate brightness level.

`HIGH` selects a full brightness level.

Examples

`DISPLAY:INTENSITY:BACKLIGHT LOW` sets the waveform backlight to low brightness level.

`DISPLAY:INTENSITY:BACKLIGHT?` might return `:DISPLAY:INTENSITY:BACKLIGHT HIGH`, indicating that the waveform backlight is set to full brightness level.

DISplay:INTENSITy:WAVEform:IMAGEView

This command sets or queries the saturation level for image view waveforms.

Group Display Control

Syntax

```
DISplay:INTENSITy:WAVEform:IMAGEView <NR2>
DISplay:INTENSITy:WAVEform:IMAGEView?
```

Related Commands DISplay:INTENSITy:WAVEform:RECORDView

Arguments `<NR2>` is the waveform saturation and ranges from 10 to 100 percent.

Examples `DISPLAY:INTENSITY:WAVEFORM:IMAGEVIEW 100` sets the image view waveform saturation to 100 percent.

DISPLAY:INTENSITY:WAVEFORM:IMAGEVIEW? might return :DISPLAY:INTENSITY:WAVEFORM:IMAGEVIEW 60.0000, indicating that the saturation level of the image view waveforms is currently set to 60 percent.

DISplay:INTENSITy:WAVEform:RECORDView

This command sets or queries the saturation level for record view waveforms.

Group Display Control

Syntax

```
DISplay:INTENSITy:WAVEform:RECORDView <NR2>
DISplay:INTENSITy:WAVEform:RECORDView?
```

Related Commands DISplay:INTENSITy:WAVEform:IMAGEView

Arguments <NR2> is the waveform saturation and ranges from 10 to 100 percent.

Examples DISPLAY:INTENSITY:WAVEFORM:RECORDVIEW 100 sets the record view waveform saturation to 100 percent.

DISPLAY:INTENSITY:WAVEFORM:RECORDVIEW? might return :DISPLAY:INTENSITY:WAVEFORM:RECORDVIEW 60.0000, indicating that the saturation of the record view waveforms is currently set to 60 percent.

DISplay:PERSistence

This command sets or queries the persistence aspect of the display. This affects the display only and is equivalent to selecting Display Persistence from the Display menu.

Group Display Control

Syntax

```
DISplay:PERSistence {OFF|INFPersist|VARpersist}
DISplay:PERSistence?
```

Related Commands DISplay:VARpersist

Arguments

`OFF` disables the persistence aspect of the display.

`INFPersist` sets a display mode where any pixels, once touched by samples, remain set until cleared by a mode change.

`VARPersist` sets a display mode where set pixels are gradually dimmed.

Examples

`DISPLAY:PERSISTENCE VARPERSIST` sets the persistence aspect of the display to fade set pixels according to the time set in the DISplay:VARpersist command.

`DISPLAY:PERSISTENCE?` might return `:DISPLAY:PERSISTENCE OFF`, indicating that the persistence aspect of the display is disabled.

DISplay:PERSistence:RESET (No Query Form)

This query-only command controls the clearing of persistence data. Persistence data includes all acquired waveform data.

This is equivalent to pressing the front panel 'Clear' button.

Group Display Control

Syntax `DISplay:PERSistence:RESET`

Related Commands DISplay:PERSistence, DISplay:VARpersist

Examples `DISPLAY:PERSISTENCE:RESET` clears the display of persistence data.

DISplay:SCREENText? (Query Only)

This query-only command returns all screen text settings.

Group Display Control

Syntax `DISplay:SCREENText?`

Related Commands DISplay:SCREENText:STATE, DISplay:SCREENText:LABel<x>?, DISplay:SCREENText:LABel<x>:NAMe, DISplay:SCREENText:LABel<x>:XPOS, DISplay:SCREENText:LABel<x>:YPOS

Examples `DISPLAY:SCREENTEXT?` might reply: `:DISPLAY:SCREENTEXT:STATE 0;LABEL1:NAME "";XPOS 100;YPOS 5;:DISPLAY:SCREENTEXT:LABEL2:NAME "";XPOS 100;YPOS 20;:DISPLAY:SCREENTEXT:LABEL3:NAME "";;XPOS 100;YPOS 35;:DISPLAY:SCREENTEXT:LABEL4:NAME "";XPOS 100;YPOS 50;:DISPLAY:SCREENTEXT:LABEL5:NAME "";XPOS 100;YPOS 65;:DISPLAY:SCREENTEXT:LABEL6:NAME "";XPOS 100;YPOS 80;:DISPLAY:SCREENTEXT:LABEL7:NAME "";XPOS 100;YPOS 95;:DISPLAY:SCREENTEXT:LABEL8:NAME "";XPOS 100;YPOS 110.`

DISplay:SCREENTExt:LABel<x>? (Query Only)

This query-only command returns the screen text setting for a given label. Specifically, it provides the name, horizontal position (XPOS), and vertical position (YPOS).

Group Display Control

Syntax `DISplay:SCREENTExt:LABel<x>?`

Related Commands DISplay:SCREENTExt?, DISplay:SCREENTExt:STATE, DISplay:SCREENTExt:LABel<x>:NAMe, DISplay:SCREENTExt:LABel<x>:XPOS, DISplay:SCREENTExt:LABel<x>:YPOS

Examples `DISPLAY:SCREENTEXT:LABEL3?` might return `:DISPLAY:SCREENTEXT:LABEL3:NAME "THIS IS SCREEN TEXT"; :XPOS 100; :YPOS 35`, indicating that the name of Label 3 is THIS IS SCREEN TEXT, and the label is position at 100 horizontally and at 35 vertically.

DISplay:SCREENTExt:LABel<x>:FONTCOlor

This command sets or queries the screen text label font color. <x> is the label number, 1 through 8.

Group Display Control

Syntax
```
DISplay:SCREENTExt:LABel<x>:FONTCOlor <QString>
DISplay:SCREENTExt:LABel<x>:FONTCOlor?
```

Arguments `<QString>` argument is the font color to be displayed for a given label.

Examples

`DISPLAY:SCREENTEXT:LABEL1:FONTCOLOR 0240120` sets the Label 1 font color to 0240120 with Hue of 0, Lightness of 240, and Saturation of 120.

`DISPLAY:SCREENTEXT:LABEL3:FONTCOlOR?` might return `:DISPLAY:SCREENTEXT:LABEL3:FONTCOlOR 16711680`, indicating that the font color of the Label 1 is set to 16711680 with Hue of 167, Lightness of 116, and Saturation of 80.

DISplay:SCREENTExt:LABel<x>:FONTNAme

This command sets or queries the screen text label font name. <x> is the label number which ranges from 1 through 8.

Group

Display Control

Syntax

```
DISplay:SCREENTExt:LABel<x>:FONTNAme <QString>
DISplay:SCREENTExt:LABel<x>:FONTNAme?
```

Arguments

`<QString>` argument is the font name to be displayed for a given label.

Examples

`DISPLAY:SCREENTEXT:LABEL1:FONTNAME ARIAL` sets the Label 1 font name to Arial.

`DISPLAY:SCREENTEXT:LABEL1:FONTNAME?` might return `DISPLAY:SCREENTEXT:LABEL1:FONTNAME "Arial Narrow"`, indicating that the Label 1 font name is set to Arial Narrow.

DISplay:SCREENTExt:LABel<x>:FONTSIze

This command sets or queries the screen text label font size. <x> is the label number which ranges from 1 through 8.

Group

Display Control

Syntax

```
DISplay:SCREENTExt:LABel<x>:FONTSIze <NR1>
DISplay:SCREENTExt:LABel<x>:FONTSIze?
```

Arguments

`<NR1>` is the font size to be displayed for a given label.

Examples `DISPLAY:SCREENTEXT:LABEL1:FONTSIZE 11` sets the Label 1 font size to 11.

`DISPLAY:SCREENTEXT:LABEL1:FONTSIZE?` might return `DISPLAY:SCREENTEXT:LABEL1:FONTSIZE 9`, indicating that the Label 1 font size is set to 9.

DISplay:SCREENTExt:LABel<x>:FONTSTyle

This command sets or queries the screen text label font style. <x> is the label number, 1 through 8.

Group Display Control

Syntax
```
DISplay:SCREENTExt:LABel<x>:FONTSTyle <QString>
DISplay:SCREENTExt:LABel<x>:FONTSTyle?
```

Arguments `<QString>` argument is the font style to be displayed for a given label.

Examples `DISPLAY:SCREENTEXT:LABEL1:FONTSTYLE BOLD` sets the Label 1 font style to BOLD.

`DISPLAY:SCREENTEXT:LABEL1:FONTSTYLE?` might return `DISPLAY:SCREENTEXT:LABEL1:FONTSTYLE "Regular"`, indicating that the Label 1 font style is Regular.

DISplay:SCREENTExt:LABel<x>:NAMe

This command sets the text to be displayed for a given label. The label is specified by <x>. The value of <x> can range from 1 through 8.

Group Display Control

Syntax
```
DISplay:SCREENTExt:LABel<x>:NAMe <QString>
DISplay:SCREENTExt:LABel<x>:NAMe?
```

Related Commands , DISplay:SCREENTExt:STATE, DISplay:SCREENTExt:LABel<x>:XPOS, DISplay:SCREENTExt:LABel<x>:YPOS

Arguments `<QString>` argument is the text to be displayed for a given label.

Examples

`DISPLAY:SCREENTEXT:LABEL2:NAME "THIS IS SCREEN TEXT"` sets the screen text for Label 2 to THIS IS SCREEN TEXT.

`DISPLAY:SCREENTEXT:LABEL2:NAME?` might return `:DISPLAY:SCREENTEXT:LABEL2:NAME "MyWaveform"`, indicating that the Label 2 screen text is set to MyWaveform.

DISplay:SCREENTExt:LABel<x>:STATE

This command sets the state to be displayed for a given label. <x> is the label number, 1 through 8.

Group

Display Control

Syntax

```
DISplay:SCREENTExt:LABel<x>:STATE {<NR1>|OFF|ON}
DISplay:SCREENTExt:LABel<x>:STATE?
```

Arguments

`OFF` = the screen text does not display.

`ON` = the screen text displays.

Examples

`DISPLAY:SCREENTEXT:LABEL2:STATE ON` sets the Label 2 screen text to display.

`DISPLAY:SCREENTEXT:LABEL2:STATE?` might return `:DISPLAY:SCREENTEXT:LABEL2:STATE 1`, indicating that the Label 2 screen text display is ON.

DISplay:SCREENTExt:LABel<x>:XPOS

This command sets the horizontal position of a given screen text label. The label is specified by x. The value of x can range from 1 through 8.

Group

Display Control

Syntax

```
DISplay:SCREENTExt:LABel<x>:XPOS <NR1>
```

Related Commands

DISplay:SCREENTExt:STATE, DISplay:SCREENTExt:LABel<x>:NAMe, DISplay:SCREENTExt:LABel<x>:YPOS

Arguments <NR1> sets the horizontal position of the screen text label. The value can range from 0 to 500.

Examples DISPLAY:SCREENTEXT:LABEL3:XPOS 100 sets the horizontal position of Label 3 to 100.

DISPLAY:SCREENTEXT:LABEL3:XPOS? might return :DISPLAY:SCREENTEXT:LABEL3:XPOS 250, indicating that the screen Label 3 is horizontally positioned to 250.

DISplay:SCREENTExt:LABel<x>:YPOS

This command sets the vertical position of a given label. The label is specified by x. The value of x can range from 1 through 8.

Group Display Control

Syntax DISplay:SCREENTExt:LABel<x>:YPOS <NR1>

Related Commands DISplay:SCREENTExt:STATE, DISplay:SCREENTExt:LABel<x>:NAMe, DISplay:SCREENTExt:LABel<x>:XPOS

Arguments <NR1> sets the vertical position of the screen text label. The value can range from 0 to 385.

Examples DISPLAY:SCREENTEXT:LABEL3:YPOS 100 sets the vertical position of Label 3 to 100.

DISPLAY:SCREENTEXT:LABEL3:YPOS? might return :DISPLAY:SCREENTEXT:LABEL3:YPOS 192, indicating that screen Label 3 is vertically positioned to 192.

DISplay:SCREENTExt:STATE

This command controls the display of screen text.

Group Display Control

Syntax DISplay:SCREENTExt:STATE {<NR1>|ON|OFF}

Related Commands DISplay:SCREENTExt?, DISplay:SCREENTExt:LABel<x>?, DISplay:SCREENTExt:LABel<x>:NAMe, DISplay:SCREENTExt:LABel<x>:XPOS, DISplay:SCREENTExt:LABel<x>:YPOS

Arguments <NR1> = 0 disables screen text; any other value enables screen text.

ON turns on the display of screen text.

OFF turns off the display of screen text.

Examples DISPLAY:SCREENTEXT:STATE ON turns the screen text display on.

DISPLAY:SCREENTEXT:STATE? might return :DISPLAY:SCREENTEXT:STATE 1, indicating that the screen text is displayed.

DISplay:SHOWREmote

NOTE. *Not available on some models.*

This command sets or queries the state of the remote display feature and is equivalent to selecting Display Remote from the Display menu. The query form of this command returns ON (1) or OFF (0).

This feature allows you to view waveforms and other graticule data on a remote display using remote control software like VNC (Virtual Network Computing) or Symantec pcAnywhere.

Group Display Control

Syntax

```
DISplay:SHOWREmote {<NR1>|ON|OFF}
DISplay:SHOWREmote?
```

Arguments <NR1> = 0 disables remote display of waveform and other graticule data; any other value enables remote display of waveform and other graticule data.

ON enables the remote display of waveform and other graticule data.

OFF disables the remote display of waveform and other graticule data.

Examples DISPLAY:SHOWREMOTE ON enables remote display of waveform and other graticule data.

DISPLAY:SHOWREMOTE? might return :DISPLAY:SHOWREMOTE 1, indicating that the remote display of waveform and other graticule data is currently enabled.

DISplay:STYle

This command sets or queries how the data is displayed for normal and FastAcq modes. This command is equivalent to selecting Display Style from the Display menu and choosing a style.

Group Display Control

Syntax

```
DISplay:STYle {DOTs|INTENSIFied|VECtors}
DISplay:STYle?
```

Arguments

`DOTs` displays individual data points. New points immediately replace old ones.

`INTENSIFied` causes the display to show interpolated samples with dark spaces (Only the "real" samples are displayed).

`VECtors` connects adjacent data points. New points immediately replace old ones.

When FastAcq mode is enabled, intensified samples display as dots. However, turning off FastAcq mode causes the display style to snap back to its previous setting.

Examples

`DISPLAY:STYLE VECTORS` sets the display to connect adjacent data points.

`DISPLAY:STYLE?` might return `:DISPLAY:STYLE INTENSIFIED`, indicating that interpolated samples are not displayed.

DISplay:TRIGBar

This command controls or queries the display of the trigger-level indicator bars. Indicator bars show where the trigger voltage level is set.

The instrument will only display the bar if the associated trigger source is also displayed. If both a main and a delayed trigger are displayed, then two bars will appear. One will accompany each source. If a logic trigger is selected, then multiple bars might appear. One will show the upper threshold and one will show the lower threshold. This command is equivalent to selecting Display Setup from the Display menu and then choosing the Objects tab.

Group Display Control

Syntax

```
DISplay:TRIGBar {OFF|SHORt|LONG}
DISplay:TRIGBar?
```

Arguments

`OFF` removes the trigger indicator bar from the display.

`SHORt` displays, as the indicator, a short arrow at the right side of the graticule for each displayed trigger signal.

`LONG` displays, as the indicator, a horizontal line across the width of the graticule for each displayed trigger signal.

Examples

`DISPLAY:TRIGBAR LONG` sets the display to show a long trigger indicator bar (or bars).

`DISPLAY:TRIGBAR?` might return `:DISPLAY:TRIGBAR OFF`, indicating that the indicator bar is removed from the display.

DISplay:TRIGT

This command controls or queries the display of the trigger T. The trigger T shows where the trigger occurred on the waveform.

Group

Display Control

Syntax

```
DISplay:TRIGT {<NR1>|OFF|ON}
DISplay:TRIGT?
```

Arguments

`<NR1>` = 0 disables the trigger T; any other value displays the trigger T.

`OFF` removes the trigger indicator T from the display.

`ON` displays a T at the trigger point.

Examples

`DISPLAY:TRIGT OFF` disables the trigger T.

`DISPLAY:TRIGT?` might return `:DISPLAY:TRIGT 1` indicating that the trigger T is displayed.

DISplay:VARpersist

This command sets or queries persistence decay time, which is the approximate decay time for a freshly struck persistence sample. This command is equivalent to selecting Display Setup from the Display menu, selecting the Appearance tab, and then entering the desired Persist Time.

Group

Display Control

Syntax

```
DISplay:VARpersist <NR3>
DISplay:VARpersist?
```

Related Commands

DISplay:PERSistence

Arguments

<NR3> indicates the persistence decay time and ranges from 0.5 to 100.

Examples

DISPLAY:VARPERSIST 5 sets the persistence decay time to 5.

DISPLAY:VARPERSIST? might return :DISPLAY:VARPERSIST 3.0000E-01, indicating that persistence decay time is currently set to 0.300.

DISplay:WAVEform

This command controls or queries the display of acquisition and reference waveforms.

Group

Display Control

Syntax

```
DISplay:WAVEform {<NR1>|OFF|ON}
DISplay:WAVEform?
```

Arguments

<NR1> = 0 removes waveforms from the display; any other value displays the waveforms.

OFF removes the waveforms from the display.

ON displays waveforms on the display.

Examples

DISPLAY:WAVEFORM OFF removes waveforms from the display.

DISPLAY:WAVEFORM? might return :DISPLAY:WAVEFORM 1, indicating that waveforms are displayed.

D<x>:LABEL

NOTE. *Not available on some models.*

This command sets or queries the label that appears for the specified digital input on the display.

Group Digital

Syntax
```
D<x>:LABEL <QString>
D<x>:LABEL?
```

Related Commands D<x>:POSition, D<x>:THRESHold

Arguments <QString> is an alphanumeric string of characters, enclosed in quotes, that defines the label text.

Examples D1:LABEL "INPUT" sets the label for the digital input D1 to INPUT.

D1:LABEL? might return :D1:LABEL "CLOCK", indicating that the label for the digital input D1 is set to CLOCK.

D<x>:POSition

NOTE. *Not available on some models.*

This command sets or queries the position for the specified digital input.

Group Digital

Syntax
```
D<x>:POSition <NR3>
D<x>:POSition?
```

Related Commands D<x>:LABEL, D<x>:THRESHold

CH<x>:POSition, D<x>:POSition, MATH<x>:VERTical:POSition

Arguments <NR3> specifies the position.

Examples D1:POSITION 2 sets the position for the digital input D1 to 2.3400 divisions.

D1:POSITION? might return :D1:POSITION 3.3400, indicating that the position for the digital input D1 is set to 3.34 divisions.

D<x>:PROBE:ID:SERnumber? (Query Only)

NOTE. *Not available on some models.*

This command queries the serial number of the digital probe that provides the specified digital signal.

Group Digital

Syntax `D<x>:PROBE:ID:SERnumber?`

Related Commands D<x>:PROBE:ID:TYPe?

Examples `D1:PROBE:ID:SERNUMBER?` might return `:D1:PROBE:ID:SERNUMBER "PQ009"`, indicating that the serial number of the digital probe that provides the specified digital signal D1 is set to PQ009.

D<x>:PROBE:ID:TYPe? (Query Only)

NOTE. *Not available on some models.*

This command queries the type of digital probe that provides the specified digital signal.

Group Digital

Syntax `D<x>:PROBE:ID:TYPe?`

Related Commands D<x>:PROBE:ID:SERnumber?

Examples `D1:PROBE:ID:TYPE?` might return `:D1:PROBE:ID:TYPE "P6717"`, indicating that the type of digital probe which provides the specified digital signal D1 is P6717.

D<x>:THRESHold

NOTE. *Not available on some models.*

This command sets or queries the threshold level for the specified digital signal.

Group Digital

Syntax

```
D<x>:THRESHold <NR3>
D<x>:THRESHold?
```

Related Commands D<x>:LABEL, D<x>:POSition,

MATH<x>:THRESHold

Arguments <NR3> specifies the threshold level in volts.

Examples D5:THRESHOLD 1.5 sets the threshold for digital channel D5 to 1.5 volts.

D14:THRESHOLD? might return :D14:THRESHOLD 1.2500 indicating that the threshold level for the digital channel D14 is set to 1.25 volts.

EMail (No Query Form)

This command (no query form) sends a test e-mail message or sets the current e-mail sent count to zero.

Group E-mail

Syntax

```
EMail {TESt|RESET}
```

Arguments TESt argument sends a test e-mail message.

RESET argument sets the e-mail sent count to zero.

Examples

```
EMAILTEST
```

This command sends a test e-mail.

EMAILRESET resets the sent mail count to zero.

EMail:ATTempts

This command sets or queries the number of times that an attempt will be made to send e-mail to the SMTP e-mail server. The default is 1.

Group E-mail

Syntax

```
EMail:ATTempts <NR1>
EMail:ATTempts?
```

Arguments <NR1> can be 1 through 20. This number is the specified number of attempts that will be made to send e-mail to the SMTP server.

Examples EMAIL:ATTEMPTS 15 specifies that 15 attempts will be made to send the e-mail to the SMTP e-mail server.

EMAIL:ATTEMPTS? might return :EMAIL:ATTEMPTS 10, indicating that 10 attempts will be made to send the e-mail to the SMTP e-mail server.

EMail:AUTHLogin

This command sets or queries the login name that will be used if the SMTP e-mail server requires one for authentication.

Group E-mail

Syntax

```
EMail:AUTHLogin {<QString>}
EMail:AUTHLogin?
```

Arguments <QString> argument is a string that specifies the login name to be used with the SMTP e-mail server.

Examples EMAIL:AUTHLOGIN kentb specifies that kentb will be used as the login name with the SMTP e-mail server.

EMAIL:AUTHLOGIN? might return :EMAIL:AUTHLOGIN "bobk", indicating that bobk will be used as the login name used with the SMTP e-mail server.

EMail:AUTHPassword (No Query Form)

This command (no query form) sets the password that will be used if the SMTP e-mail server requires one for authentication.

Group E-mail

Syntax
```
EMail:AUTHPassword {<QString>}
```

Arguments <QString> argument is a string that specifies the password to be used with the SMTP e-mail server.

Examples EMAIL:AUTHPASSWORD "123xyz" specifies that 123xyz will be used as the password with the SMTP e-mail server.

EMail:COUNt? (Query Only)

This query only command returns the number of e-mails that have been sent since Email on Event was armed. The range of values returned can be from 0 to 50.

Group E-mail

Syntax
```
EMail:COUNt?
```

Examples EMAIL:COUNT? might return :EMAIL:COUNT 15, indicating that 15 e-mails have been sent since e-mail on event was armed.

EMail:FROm

This command sets or queries the From line in the e-mail. The default for the From line is the instrument model and serial number.

Group E-mail

Syntax
```
EMail:FROm {<QString>}
EMail:FROm?
```

Arguments <QString> argument is a string that is placed in the From line of the e-mail. An example is “johnz”.

Examples EMAIL:FROM "kentb" specifies that kentb will be placed in the From line of the e-mail.

EMAIL:FROM? might return :EMAIL:FROM "BOBK", indicating that bobk will appear in the From line of the e-mail.

EMail:HOSTwanted

This command sets or queries the host name that will be used when e-mail is sent to the SMTP e-mail server if the DPO host name will not work.

Group E-mail

Syntax
```
EMail:HOSTwanted {<QString>}
EMail:HOSTwanted?
```

Arguments <QString> argument is a string that specifies the host name to be used with the SMTP e-mail server if the default DPO host name will not work.

Examples EMAIL:HOSTWANTED "LazyEight" specifies that LazyEight will be used as the host name when sending e-mail to the SMTP e-mail server.

EMAIL:HOSTWANTED? might return :EMAIL:HOSTWANTED "LazyEight", indicating that LazyEight is the host name used with the SMTP e-mail server.

EMail:IMAGe

This command sets or queries whether a screen image is included in e-mail. If this command is set to ON, the format and content of the screen image included is specified using the EXPort commands within the Hard Copy Command group.

Group E-mail

Syntax
```
EMail:IMAGe {<NR1>|ON|OFF|}
EMail:IMAGe?
```

Related Commands EXPort

Arguments

<NR1> = 0 disables the inclusion of a screen image in e-mail; any other value enables the inclusion of a screen image in e-mail.

ON argument enables the inclusion of a screen image in e-mail.

OFF argument disables the inclusion of a screen image in e-mail.

Examples

EMAIL:IMAGE 1 enables the inclusion of a screen image in e-mail.

EMAIL:IMAGE? might return :EMAIL:IMAGE 0 indicating that a screen image is not included in the e-mail.

EMail:MASK

This command sets or queries whether e-mail is sent when a mask test failure occurs.

Group

E-mail

Syntax

```
EMail:MASK {<NR1>|ON|OFF}
EMail:MASK?
```

Arguments

<NR1> = 0 disables sending e-mail when a mask test failure occurs; any other value enables sending e-mail when a mask test failure occurs.

ON argument enables sending e-mail when a mask test failure occurs.

OFF disables sending e-mail when a mask test failure occurs.

Examples

EMAIL:MASK 1 enables sending e-mail when a mask test failure occurs.

EMAIL:MASK? might return :EMAIL:MASK 0, indicating that e-mail will not be sent when a mask test failure occurs.

EMail:MAXSize

This command sets or queries the maximum size (in megabytes) of e-mail that can be sent to the SMTP server.

Group

E-mail

Syntax

```
EMail:MAXSize {<NR1>}
EMail:MAXSize?
```

Arguments <NR1> can be 0 through 2000 (megabytes). This is the maximum size of each e-mail that can be sent to the SMTP server.

Examples EMAIL:MAXSIZE 5 sets the maximum size of each e-mail enclosure to 5 megabytes of data.

EMAIL:MAXSIZE? might return :EMAIL:MAXSIZE 10, indicating that each e-mail sent to the SMTP server can contain up to 10 megabytes of data.

EMail:MEASUrement

This command sets or queries whether measurement data is included as an attachment to e-mail. You must specify the saved measurement selection and the measurement format using the Email on Event setup menu. There are no remote commands for this purpose.

Group E-mail

Syntax
```
EMail:MEASUrement {<NR1>|ON|OFF}
EMail:MEASUrement?
```

Arguments <NR1> = 0 disables the inclusion of measurement data in the e-mail; any other value enables including measurement data in the e-mail.

ON enables the inclusion of measurement data in e-mail.

OFF disables the inclusion of measurement data in e-mail.

Examples EMAIL:MEASUREMENT 1 enables the inclusion of measurement data in e-mail.

EMAIL:MEASUREMENT? might return :EMAIL:MEASUREMENT 0, indicating that measurement data is not included in the e-mail.

EMail:NUMEMails

This command sets or queries the number of e-mails you can send when Email on Event is armed, from 1 to 50. This limit is to restrict e-mail floods.

Group E-mail

Syntax
```
EMail:NUMEMails {<NR1>}
EMail:NUMEMails?
```

Related Commands EMail

Arguments <NR1> can be 1 through 50. This is the number of e-mails you can send before the number of sent e-mails must be reset with the EMail RESet command.

Examples EMAIL:NUMEMAILS 10 sets the number of e-mails that can be sent to 10.

EMAIL:NUMEMAILS? might return :EMAIL:NUMEMAILS 5, indicating that five e-mails can be sent.

EMail:SMTPPort

This command sets or queries the SMTP port number that the e-mail server uses if other than the default of 25.

Group E-mail

Syntax
```
EMail:SMTPPort {<NR1>}
EMail:SMTPPort?
```

Arguments <NR1> can be 1 through 65535. This number specifies the TCPIP port number.

Examples EMAIL:SMTPPORT 55 specifies that the SMTP e-mail server will communicate on TCPIP port 55.

EMAIL:SMTPPORT? might return :EMAIL:SMTPPORT 10, indicating that TCPIP port 10 will be used to communicate with the SMTP e-mail server.

EMail:SMTPServer

This command sets or queries the address of the SMTP mail server.

Group E-mail

Syntax
```
EMail:SMTPServer <QString>
EMail:SMTPServer?
```

Arguments <QString> argument is the address of the SMTP mail server that will handle the mail service request. For example, “smtp.tek.com”.

Examples `EMAIL:SMTPSERVER "smtp.tek.com"` sets the SMTP mail server to smtp.tek.com. This will be the mail server used for mail service requests.

`EMAIL:SMTPSERVER?` might return `:EMAIL:SMTPSERVER "SMTP.TEK.COM"`, indicating that the SMTP mail server is set to SMTP.tek.com.

EMail:STATUS? (Query Only)

This query only command returns the status of the last e-mail you attempted to send.

Group E-mail

Syntax

```
EMail:STATUS?
```

Returns The status might be one of the following strings:

- PASS - indicates that the last e-mail message sent was successful or that no email message was attempted.
- FAIL - indicates that the last e-mail message attempted was not sent.
- SEND - indicates that an e-mail message is currently being sent.
- HOST_FAIL - indicates that the instrument failed to connect to the SMTP server.
- CONNECTION_FAIL - indicates that the SMTP server refused the connection.

Examples `EMAIL:STATUS?` might return `:EMAIL:STATUS FAIL`, indicating that the e-mail failed to be sent.

EMail:TIMEOut

This command sets or queries the global timeout in seconds. The default is 30 seconds. You use this timeout for socket connections and might need to change it from the default on some networks.

Group E-mail

Syntax

```
EMail:TIMEOut <NR1>
EMail:TIMEOut?
```

Arguments <NR1> argument is the global timeout in seconds. The value for NR1 can be 0 through 500.

Examples EMAIL:TIMEOUT 60 that 60 seconds will be used for network connection timeouts.

EMAIL:TIMEOUT? might return :EMAIL:TIMEOUT 30, indicating that the default of 30 seconds is set for network connection timeouts.

EMail:TO

This command sets or queries the address of the recipient(s) of an e-mail.

Group E-mail

Syntax
```
EMail:TO <QString>
EMail:TO?
```

Arguments <QString> argument is the e-mail address of the recipient (or recipients). Multiple addresses are separated with semicolons (;). For example, “johndoe@tek.com;billsmith@tek.com” specifies that both johndoe and billsmith at tek.com will receive e-mail when the event occurs. The length of this string is limited to 252 characters.

Examples EMAIL:TO "johndoe@tek.com" will send e-mail to johndoe@tek.com when the event occurs.

EMAIL:TO? might return :EMAIL:TO "billsmill@tek.com", indicating that the billsmith@tek.com will receive e-mail when the event occurs.

EMail:TRIGger

This command sets or queries whether e-mail is sent when a trigger occurs. The e-mail is sent after the acquisition is complete.

Group E-mail

Syntax
```
EMail:TRIGger {<NR1>|ON|OFF}
EMail:TRIGger?
```

Arguments

<NR1> = 0 disables sending e-mail when a trigger occurs; any other value enables sending e-mail when a trigger occurs.

ON argument enables sending e-mail when a trigger occurs.

OFF disables sending e-mail when a trigger occurs.

Examples

EMAIL:TRIGGER 1 enables sending e-mail when a trigger occurs.

EMAIL:TRIGGER? might return :EMAIL:TRIGGER 0, indicating that e-mail will not be sent when a trigger occurs.

EMail:WAVEform

This command sets or queries whether waveform data is included in e-mail. The waveforms to be included, the start and stop points (and, or, frames for Fast Frame mode), and the data formats are specified by the DATa and WFMOutpre? commands within the Waveform Transfer Command group.

Group

E-mail

Syntax

```
EMail:WAVEform {<NR1>|ON|OFF}
EMail:WAVEform?
```

Arguments

<NR1> = 0 disables the inclusion of waveform data in the e-mail; any other value enables including waveform data in the e-mail.

ON enables the inclusion of waveform data in e-mail.

OFF disables the inclusion of waveform data in e-mail.

Examples

EMAIL:WAVEFORM 1 enables the inclusion of waveform data in e-mail.

EMAIL:WAVEFORM? might return :EMAIL:WAVEFORM 0, indicating that waveform data is not included in the e-mail.

ERRORDetector:ALERT

This command sets or queries the error detector alert.

Group

Error Detector

Syntax

```
ERRORDetector:ALERT {ON|OFF|<NR1>}
ERRORDetector:ALERT?
```

Arguments

ON turns on the error detector alert.

OFF turns off the error detector alert.

<NR1> = 0 disables the alert; any other value enables the alert.

Examples

ERRORDETECTOR:ALERT? might return :ERRORDETECTOR:ALERT OFF indicating the error detector alert is off.

ERRORDetector:ALIGNCHARacter? (Query Only)

This command queries all of the align character values. Align characters must be defined for all test types, and those characters must appear in the signal test pattern. The Align character may be in symbolic or bit string form

NOTE. *Not available on some models.*

Group

Error Detector

Syntax

```
ERRORDetector:ALIGNCHARacter?
```

Examples

ERRORDETECTOR:ALIGNCHARACTER? might return :ERRORDETECTOR:ALIGNCHARACTER:SYMBOL “K28.5”; PLUS"1100000101";MINUS"0011111010"; to indicate that the symbolic K28.5 Align Character is also represented by the RD+ bit string value is 1100000101 and the RD- align character bit string value 0011111010.

ERRORDetector:ALIGNCHARacter:MINus? (Query Only)

This command queries the RD- align character bit string value. Reception of this character by the instrument causes the receiver to align to the 10-bit character boundary.

NOTE. *Not available on some models.*

Group

Error Detector

Syntax `ERRORDetector:ALIGNCHARacter:MINus?`

Examples `ERRORDETECTOR:ALIGNCHARACTER:MINUS??` might return `ERRORDETECTOR:ALIGNCHARACTER:MINUS "0011111010"`, indicating that the RD- align bit string value is "0011111010".

ERRORDetector:ALIGNCHARacter:PLUS? (Query Only)

This command queries the RD+ align character bit string value. Reception of the align character by the instrument aligns the receiver to the 10-bit character boundary.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:ALIGNCHARacter:PLUS??`

Examples `ERRORDETECTOR:ALIGNCHARACTER:PLUS??` might return `ERRORDETECTOR:ALIGNCHARACTER:PLUS "1100000101"`, indicating that the RD+ align character value is 1100000101.

ERRORDetector:ALIGNCHARacter:SYMBOL

This command sets or queries the symbolic align character value. Reception of the align character by the instrument aligns the receiver to the 10-bit character boundary.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:ALIGNCHARacter:SYMBOL <QString>
ERRORDetector:ALIGNCHARacter:SYMBOL?
```

Arguments `<QString>` is a quoted string representing a symbolic character, e.g., "K28.5".

Examples ERRORDETECTOR:ALIGNCHARACTER:SYMBOL “K28.5” sets that the symbolic align character to “K28.5”.

ERRORDETECTOR:ALIGNCHARACTER:SYMBOL? might return ERRORDETECTOR:ALIGNCHARACTER:SYMBOL “K28.5”, indicating that the symbolic align character value is K28.5.

ERRORDetector:ALIGNPRIMitive? (Query Only)

This command queries all of the align primitive values, including its state (ON=1/OFF=0), the length four array of symbolic character values, and the RD+ and RD- .length four arrays of bit string values.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:ALIGNPRIMitive?

Examples ERRORDETECTOR:ALIGNPRIMITIVE? might return ERRORDETECTOR:ALIGNPRIMITIVE:STATE 1;SYMBOL1 "K28.5";SYMBOL2 "D10.2";SYMBOL3 "D10.2";SYMBOL4 "D27.3";PLUS1 "1100000101";PLUS2 "0101010101";PLUS3 "0101010101";PLUS4 "0010011100";MINUS1 "0011111010";MINUS2 "0101010101";MINUS3 "0101010101";MINUS4 "1101100011"

ERRORDetector:ALIGNPRIMitive:MINUS? (Query Only)

This command queries the RD- align primitive bit string values.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:ALIGNPRIMitive:MINUS?

Examples ERRORDETECTOR:ALIGNPRIMITIVE:MINUS? might return ERRORDETECTOR:ALIGNPRIMITIVE:MINUS “001111010”

ERRORDetector:ALIGNPRIMitive:MINus<x>? (Query Only)

This command queries the RD- align primitive bit string values.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:ALIGNPRIMitive:MINus<x>?`

Examples `ERRORDetector:ALIGNPRIMitive:MINus1?` might return `:ERRORDETECTOR:ALIGNPRIMITIVE:MINUS1 "0011111010"` .

`ERRORDetector:ALIGNPRIMitive:MINus2?` might return `:ERRORDETECTOR:ALIGNPRIMITIVE:MINUS2 "0101010101"`.

ERRORDetector:ALIGNPRIMitive:PLUS? (Query Only)

This command queries the four RD+ align primitive bit string values.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:ALIGNPRIMitive:PLUS?`

Examples `ERRORDETECTOR:ALIGNPRIMITIVE:PLUS?` might return “1100000101”.

`ERRORDETECTOR:ALIGNPRIMITIVE:PLUS?` might return `:ERRORDETECTOR:ALIGNPRIMITIVE:PLUS “1100000101”`, indicating the first align primitive bit string is set to that bit string value.

ERRORDetector:ALIGNPRIMitive:PLUS<x>? (Query Only)

This command queries the align primitive plus value.

Group Error Detector

Syntax `ERRORDetector:ALIGNPRIMitive:PLUS<x>?`

Related Commands

Examples `ERRORDetector:ALIGNPRIMitive:PLUS1?` might return `:ERRORDETECTOR:ALIGNPRIMITIVE:PLUS1 "1100000101"` indicating the first align primitive plus1 value is `"1100000101"`.

ERRORDetector:ALIGNPRIMitive:STATE

This command sets or queries the status of the RD- align primitive option.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:ALIGNPRIMitive:STATE {<NR1>|OFF|ON}
ERRORDetector:ALIGNPRIMitive:STATE?
```

Arguments `ON` enables the align primitive option.

`OFF` disables the align primitive option.

`<NR1>` = 0 disables the align primitive option; any other value enables the option

Examples `ERRORDETECTOR:ALIGNPRIMITIVE:STATE` ON enables the RD- align primitive option.

`ERRORDETECTOR:ALIGNPRIMITIVE:STATE?` might return `ERRORDETECTOR:ALIGNPRIMITIVE:STATE ON`, indicating that the align primitive option is enabled.

ERRORDetector:ALIGNPRIMitive:SYMBOLS

Sets or queries the four align primitive symbols. You can set one or more symbols with the command, but they must be done in order. The query returns all four symbols.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:ALIGNPRIMitive:SYMBOLS
ERRORDetector:ALIGNPRIMitive:SYMBOLS?
```

Arguments

<Qstring> is a quoted string representing one of the four align primitive symbols, such as “K28.5”. You can set one or more of the symbols with a single command, but the symbols must be set in order.

Examples

ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOLS “K28.5”,”D10.2”,”D10.2”,”D27.3” sets the four align primitive symbols.

ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOLS “K28.5”,”D10.2”,”D10.2” sets the only the first three align primitive symbols.

ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOLS? might return :ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOLS “K28.5”,”D10.2”,”D10.2”,”D27.3”, indicating the four symbols comprising the align primitive.

ERRORDetector:ALIGNPRIMitive:SYMBOL<x>

Sets or queries the align primitive symbol. The individual symbolic array elements may be accessed through SYMbol1, SYMBol2, etc.

NOTE. *Not available on some models.*

Group

Error Detector

Syntax

```
ERRORDetector:ALIGNPRIMitive:SYMBOL<x> <QString>
ERRORDetector:ALIGNPRIMitive:SYMBOL<x>?
```

Arguments

<QString> is a quoted string representing the symbolic align primitive symbols such as K28.5.

Examples

ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOL “K28.5” sets the first symbolic align primitive symbol to K28.5.

ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOL2 “D10.2” sets the second align primitive symbol.

ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOL? might return :ERRORDETECTOR:ALIGNPRIMITIVE:SYMBOL “K28.5”, indicating that the first symbolic align primitive symbol is set to K28.5.

ERRORDetector:BIT:LENgth

This command sets or queries the signal test pattern length in bits for non-USB bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:BIT:LENgth <NR1>
ERRORDetector:BIT:LENgth?
```

Arguments <NR1> indicates the bit length of the signal test pattern.

Examples ERRORDETECTOR:BIT:LENGTH 800 sets the signal test pattern length to 800.

ERRORDETECTOR:BIT:LENGTH? might return :ERRORDETECTOR:BIT:LENGTH 800, indicating that the signal test pattern length is 800.

ERRORDetector:BIT:SYNCPATtern? (Query Only)

This command queries all of the sync pattern forms and associated settings for non-USB bit error testing. The SYNCPATtern consists of one to four symbolic 10-bit 8B10B characters and their matching RD+ and RD- bit string equivalents. It is required for all non-USB bit error testing. This sequence of 8B10B characters must actually occur and be unique in the signal test pattern sent to the oscilloscope for error detection to work correctly. The SyncPattern has a symbolic form ("K28.5"), an RD+ and RD- bit string form ("0011111010"), and an as-used bit string form. The actual form used depends on the SyncPattern:Disparity and SyncPattern:Advanced settings. When Advanced is Off, the disparity setting determines the disparity of the first (leftmost) syncpattern element, and the others alternate disparity. When Advanced is On, the Disparity is set for each SyncPattern element.

Group Error Detector

Syntax

```
ERRORDetector:BIT:SYNCPATtern?
```

Examples

```
ERRORDETECTOR:BIT:SYNCPATTERN??  might
return ERRORDETECTOR:BIT:SYNCPATTERN
'"0011010100110100110100110101001101001100";"D12.0","D11.4","D12.0","D1
```

indicating that the 10-, 20-, 30-, or 40-bit sync pattern (bit string form) in use is "00110101001101001101001101010011010011 00", the symbolic four-character syncpattern (D12.0, D1.4, D12.0, D11.3), the RD+ and RD- bit string versions of the same four characters, the Disparity settings for each sycn pattern element, and the advanced enable control (which allows each character to have its own disparity setting), which is off in this example. These parameters can be queried separately using PI commands that follow.

ERRORDetector:BIT:SYNCPATtern:ADVanced

This command sets or queries the SYNCPATtern:ADVanced setting. When Advanced is ON, the Disparity of each SYNCPATtern element is controlled individually. When Advanced is off, the SYNCPATtern: DISParity alternates based on the DISParity of the first element.

Group Error Detector

Syntax

```
ERRORDetector:BIT:SYNCPATtern:ADVanced {<NR1>|OFF|ON}
ERRORDetector:BIT:SYNCPATtern:ADVanced?
```

Arguments

ON allows the Disparity of each SYNCPATtern element to be controlled individually

OFF alternates the SYNCPATtern: DISParity based on the DISParity of the first element.

<NR1> = 0 disables the advanced sync pattern; any other value enables the advanced sync pattern.

Examples

ERRORDETECTOR:BIT:SYNCPATTERN:ADVANCED? might return
ERRORDETECTOR:BIT:SYNCPATTERN:ADVanced OFF;

indicating the Advanced (Disparity feature) is OFF.

ERRORDetector:BIT:SYNCPATtern:BITString

This command queries the 10-, 20-, 30-, or 40-bit sync pattern in bit string form.

Group Error Detector

Syntax

```
ERRORDetector:BIT:SYNCPATtern:BITString <QString>
ERRORDetector:BIT:SYNCPATtern:BITString?
```

Arguments <QString> is the bit string.

Examples ERRORDETECTOR:BIT:SYNCPATTERN:BITSTRING? might return ERRORDETECTOR:BIT:SYNCPATTERN:BITSTRING "0011010100110100110100110101001101001100"; indicating the (max 40-bit) sync pattern in bit string form. The sync pattern should always be given in multiples of 10 bits.

ERRORDetector:BIT:SYNCPATtern:DISParity<x>

Sets or queries the four sync pattern Disparity elements, when the SYNCpat:ADVanced is ON. When Advanced is off, the DISParity alternates based on the DISParity of the first element.

Group Error Detector

Syntax
```
ERRORDetector:BIT:SYNCPATtern:DISParity<x> {RDPLUS | RDMINUS}
ERRORDetector:BIT:SYNCPATtern:DISParity<x>?
```

Arguments RDPLUS sets the sync pattern disparity element to RDPLUS.

RDMINUS sets the sync pattern disparity element to RDMINUS.

Examples ERRORDETECTOR:BIT:SYNCPATTERN:DISParity? might return ERRORDETECTOR:BIT:SYNCPATTERN:DISPARITY RDPLUS, RDMINUS, RDPLUS, RDMINUS;

indicating the values of the four sync pattern Disparity elements. The PI array syntax is used to access the elements individually (ERRORDetector:BIT:SYNCPATtern:DISParity2 RDPLUS, ERRORDetector:BIT:SYNCPATtern:DISParity3 RDMINUS, etc.).

ERRORDetector:BIT:SYNCPATtern:MINus<x>? (Query Only)

This command queries the four RD- bit string sync pattern elements.

Group Error Detector

Syntax ERRORDetector:BIT:SYNCPATtern:MINus<x>?

Examples ERRORDETECTOR:BIT:SYNCPATTERN:MINUS? might return ERRORDETECTOR:BIT:SYNCPATTERN:MINUS "0011011011","1101001101","0011011011","1101001100"; indicating the values for the RD- bit string sync pattern elements. An element may be queried as "X" to indicating "don't care." The PI array syntax is used to access the elements individually (ERRORDetector:BIT:SYNCPATtern:MINUS1, ERRORDetector:BIT:SYNCPATtern:MINUS2, etc.).

ERRORDetector:BIT:SYNCPATtern:PLUS<x>? (Query Only)

This command queries the four RD+ bit string sync pattern elements.

Group Error Detector

Syntax ERRORDetector:BIT:SYNCPATtern:PLUS<x>?

Examples ERRORDETECTOR:BIT:SYNCPATTERN:PLUS? might return ERRORDETECTOR:BIT:SYNCPATTERN:PLUS "0011010100","1101000010","0011010100","1101000011"; indicating the values for the RD+ bit string sync pattern elements. An element may be queried as "X" indicating "don't care." The PI array syntax is used to access the elements individually (ERRORDetector:BIT:SYNCPATtern:PLUS1?, ERRORDetector:BIT:SYNCPATtern:PLUS2?, etc.).

ERRORDetector:BIT:SYNCPATtern:SYMBOLS

Sets or queries the four sync pattern symbols. You can set one or more symbols with the command, but they must be done in order. The query returns all four symbols. A symbol may be set or queried as "X" to indicate "don't care."

Group Error Detector

Syntax
```
ERRORDetector:BIT:SYNCPATtern:SYMBOLS <Qstring>
ERRORDetector:BIT:SYNCPATtern:SYMBOLS?
```

Arguments <Qstring> is a string representing one or more symbols.

Examples ERRORDETECTOR:BIT:SYNCPATTERN:SYMBOLS? might return ERRORDETECTOR:BIT:SYNCPATTERN:SYMBOLS

"D12.0","D11.4","D12.0","D11.3"; indicating the four symbols comprising the sync pattern.

ERRORDetector:BIT:SYNCPATtern:SYMBOL<x>

This command sets or queries the four symbolic sync pattern elements.

Group Error Detector

Syntax
```
ERRORDetector:BIT:SYNCPATtern:SYMBOL<x> <Qstring>
ERRORDetector:BIT:SYNCPATtern:SYMBOL<x>?
```

Examples ERRORDETECTOR:BIT:SYNCPATTERN:SYMBol? might return ERRORDETECTOR:BIT:SYNCPATTERN:SYMBol "D12.0","D11.4","D12.0","D11.3";

indicating the values for the symbolic sync pattern elements. An element may be set or queried as X to indicate don't care. The PI array syntax is used (SYMBol1, SYMBol2, etc.).

ERRORDetector:BIT:TEST (No Query Form)

This command initiates and terminates bit error testing for the arguments START and STOP. It zeroes bit error test results for the argument CLEAR. Also copies the test pattern from the signal to memory for the argument LEARN. It re-synchronizes the recovered clock for argument SYNC.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:BIT:TEST <LEARN|START|STOP|CLEAR|SYNC>
```

Arguments LEARN copies the test pattern from the signal to the hardware template memory.

START initiates bit error counting of bit errors and duration.

STOP terminates bit error counting of bit errors and duration.

CLEAR zeroes the bit error count, duration, and bit error rate.

SYNC resynchronizes the recovered clock.

Examples `ERRORDETECTOR:BIT:TEST LEARN` copies the test pattern from the signal to the hardware template memory.

ERRORDetector:BIT:TEST:COUNt? (Query Only)

This command queries the bit error count for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:COUNt?`

Examples `ERRORDETECTOR:BIT:TEST:COUNT?` might return `:ERRORDetector:BIT:TEST:COUNT 10`, indicating that the bit error count for bit error testing is 10.

ERRORDetector:BIT:TEST:DURATION? (Query Only)

This command queries the elapsed duration (in units of bits) tested for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:DURATION?`

Examples `ERRORDETECTOR:BIT:TEST:DURATION?` might return `ERRORDetector:BIT:TEST:DURATION 7612`, indicating that the elapsed duration (in units of bits) for bit error testing is 7612.

ERRORDetector:BIT:TEST:MAXALIGNS? (Query Only)

This command queries the maximum consecutive SATA align primitives or USB skip order sets for bit error testing. The maximum number of align primitives is a design parameter of the bus standard.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:MAXALIGNS?`

Examples `ERRORDETECTOR:BIT:TEST:MAXALIGNS?` might return `:ERRORDETECTOR:BIT:TEST:MAXALIGNS 3`, indicating that the maximum consecutive SATA align primitives or USB skip order sets encountered for bit error testing is 3.

ERRORDetector:BIT:TEST:RATE? (Query Only)

This command queries the calculated bit error rate for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:RATE?`

Examples `ERRORDETECTOR:BIT:TEST:RATE?` might return `:ERRORDETECTOR:BIT:TEST:RATE 1.389e-13`, indicating that the calculated bit error rate for bit error testing is 1.389e-13.

ERRORDetector:BIT:TEST:RESults? (Query Only)

This command queries all of the results for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:RESults?`

Returns `COUNT` which is the bit error count (number of bad bits).

DURATION which is the elapsed duration (number of bits tested).

RATE which is the calculated bit error rate.

MAXALIGNS which is the maximum consecutive aligns encountered.

SECONDS which is the elapsed time in seconds.

TIME which is the elapsed duration time in Days, Hours, Minutes, and Seconds.

Examples ERRORDETECTOR:BIT:TEST:RESULTS? might return :ERRORDETECTOR:BIT:TEST:RESULTS 2,100000000000,0.2 e-12,3,15,0,0,0,15, indicating that the bit error count is 2, elapsed duration is 100000000000, calculated bit error rate is 0.2 e-12, maximum consecutive aligns encountered is 3, elapsed time is 15 seconds, and the elapsed time is 0 days, 0 hours, 0 minutes, and 15 seconds for bit error testing.

ERRORDetector:BIT:TEST:SECOnds? (Query Only)

This command queries the elapsed time in seconds for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:BIT:TEST:SECOnds?

Examples ERRORDETECTOR:BIT:TEST:SECONDS? might return :ERRORDETECTOR:BIT:TEST:SECONDS 30, indicating that the elapsed time is 30 seconds.

ERRORDetector:BIT:TEST:STATUS? (Query Only)

This command queries all of the bit error test status bits.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:BIT:TEST:STATUS?

Returns COUNTING or NOT_COUNTING for START status for bit error testing.

LOCKED or NOT_LOCKED for LOCK status for bit error testing.

SYNC or NOT_SYNC for SYNC status for bit error testing.

SIGNAL or NO_SIGNAL for SIGNAL status for bit error testing.

MAX_AP_OK or MAX_AP_FAIL for MAX_AP status for bit error testing.

Examples ERRORDETECTOR:BIT:TEST:STATUS? might return :ERRORDETECTOR:BIT:TEST:STATUS STOPPED;LOCKED;NOT_SYNC;NO_SIGNAL;MAX_AP_OK, indicating that the bits error status are STOPPED, LOCKED, NOT_SYNC, NO_SIGNAL, and MAX_AP_OK.

ERRORDetector:BIT:TEST:STATUS:LOCK? (Query Only)

This command queries the LOCK status for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:BIT:TEST:STATUS:LOCK?

Returns LOCKED or NOT_LOCKED for LOCK status for bit error testing.

Examples ERRORDETECTOR:BIT:TEST:STATUS:LOCK? might return :ERRORDETECTOR:BIT:TEST:STATUS:LOCK LOCKED, indicating that the recovered clock reference signal is LOCKED.

ERRORDetector:BIT:TEST:STATUS:MAX_AP? (Query Only)

This command queries the MAX_AP status for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:BIT:TEST:STATUS:MAX_AP?

Returns MAX_AP_OK or MAX_AP_FAIL for MAX_AP status for bit error testing.

Examples ERRORDETECTOR:BIT:TEST:STATUS:MAX_AP? might return :ERRORDETECTOR:BIT:TEST:STATUS:MAX_AP MAX_AP_OK, indicating that the status of MAX_AP is ok.

ERRORDetector:BIT:TEST:STATUS:SIGNAL? (Query Only)

This command queries the SIGNAL status for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:BIT:TEST:STATUS:SIGNAL?

Returns SIGNAL or NO_SIGNAL for SIGNAL status for bit error testing.

Examples ERRORDETECTOR:BIT:TEST:STATUS:SIGNAL? might return :ERRORDETECTOR:BIT:TEST:STATUS:SIGNAL SIGNAL, indicating that the signal is connected for bit error testing.

ERRORDetector:BIT:TEST:STATUS:START? (Query Only)

This command queries the START status for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:BIT:TEST:STATUS:START?

Returns COUNTING or NOT_COUNTING for START status for bit error testing.

Examples ERRORDETECTOR:BIT:TEST:STATUS:START? might return :ERRORDETECTOR:BIT:TEST:STATUS:START COUNTING, indicating that the bit error comparisons are being made.

ERRORDetector:BIT:TEST:STATUS:SYNC? (Query Only)

This command queries the SYNC status for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:STATUS:SYNC?`

Returns `SYNC` or `NOT_SYNC` for `SYNC` status for bit error testing.

Examples `ERRORDETECTOR:BIT:TEST:STATUS:SYNC?` might return `:ERRORDETECTOR:BIT:TEST:STATUS:SYNC NOT_SYNC`, indicating the status of the SYNC for bit error testing.

ERRORDetector:BIT:TEST:TIME? (Query Only)

This command queries the elapsed time (in days, hours, minutes, and seconds) for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:TIME?`

Examples `ERRORDETECTOR:BIT:TEST:TIME?` might return `:ERRORDETECTOR:BIT:TEST:TIME 0,0,3,14`, indicating that the elapsed time is 0 days, 0 hours, 3 minutes, and 14 seconds for bit error testing.

ERRORDetector:BIT:TEST:TIME:DAYS? (Query Only)

This command queries the elapsed time days component for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:TIME:DAYS?`

Examples `ERRORDETECTOR:BIT:TEST:TIME:DAYS?` might return `:ERRORDETECTOR:BIT:TEST:TIME:DAYS 0`, indicating that the elapsed time days component is 0 for bit error testing.

ERRORDetector:BIT:TEST:TIME:HOURS? (Query Only)

This command queries the elapsed time hours component for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:TIME:HOURS?`

Examples `ERRORDETECTOR:BIT:TEST:TIME:HOURS?` might return `:ERRORDETECTOR:BIT:TEST:TIME:HOURS 0`, indicating that the elapsed time hours component is 0 for bit error testing.

ERRORDetector:BIT:TEST:TIME:MINUTES? (Query Only)

This command queries the elapsed time minutes component for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:BIT:TEST:TIME:MINUTES?`

Examples `ERRORDETECTOR:BIT:TEST:TIME:MINUTES?` might return `:ERRORDETECTOR:BIT:TEST:TIME:MINUTES 0`, indicating that the elapsed time minutes component is 0 for bit error testing.

ERRORDetector:BIT:TEST:TIME:SECOnds? (Query Only)

This command queries the elapsed time seconds component for bit error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:BIT:TEST:TIME:SECOnds?
```

Examples `ERRORDETECTOR:BIT:TEST:TIME:SECONDS?` might return `:ERRORDETECTOR:BIT:TEST:TIME:SECONDS 20`, indicating that the elapsed time seconds component for bit error testing is 20.

ERRORDetector:BITRate

This command sets or queries error detector bitrate enumeration for error detection. There are two bitrate enumerations for each standard: 1) The standard bitrate, say RATE6000, meaning 6.0Gb/s, for SATA Gen3; and 2) Custom. When Custom is selected, the ERRORDetector:BITRATE:VALUE must also be set to the specific desired bitrate, say 6.1 Gb/s.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:BITRate {RATEcustom:CUSTOM
|RATE312000000:RATE312 |RATE1250000000:RATE1250
|RATE1500000000:RATE1500 |RATE2125000000:RATE2125
|RATE2500000000:RATE2500 |RATE3000000000:RATE3000
|RATE3125000000:RATE3125 |RATE4250000000:RATE4250
|RATE5000000000:RATE5000 |RATE6000000000:RATE6000
|RATE6250000000:RATE6250}
ERRORDetector:BITRate?
```

Arguments `RATE. . . :RATE . . .` sets the error detector bit rate to the specified value.

RATEcustom:CUSTOM allows you to set a custom bit rate using the `:ERRORDETECTOR:BITRATE:VALUE` command.

Examples `ERRORDETECTOR:BITRATE RATE6000` sets the error detector bit rate to 6.0 Gb/s.

`ERRORDETECTOR:BITRATE?` might return `:ERRORDETECTOR:BITRATE RATE6000` indicating the bitrate is set to 6.0000E+9.

ERRORDetector:BITRate:VALue

This command sets or queries error detector custom bitrate value for error detection. To set the custom value, you must also set ERRORDetector:BITTRATE to CUSTOM. The bitrate range is nominally 1.25Gb/s to 6.25 Gb/s. Special coding also allows the custom bitrate to range from 200Mb/s to 350Mb/s for PRBS7 and PRBS9 only.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:BITRate:VALue <NR3>
ERRORDetector:BITRate:VALue?
```

Arguments `<NR3>` is the custom bit rate value.

Examples `ERRORDETECTOR:BITRATE:VALUE 6.5E+9` set the custom bitrate value to 6.5 Gb/s.

`ERRORDETECTOR:BITRATE:VALUE?` might return `:ERRORDETECTOR:BITRATE:VALUE 6.0000E+9` indicating the bitrate value is set to 6.0000E+9.

ERRORDetector:CHANnel

Sets or queries error detector channel selection.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:CHANnel {CH1 | CH2 | CH3 | CH4}
ERRORDetector:CHANnel?
```

Arguments CHx is the error detector channel selection.

Examples ERRORDETECTOR:CHANNEL CH1 sets the error detector channel to CH1.

ERRORDETECTOR:CHANNEL? might return :ERRORDETECTOR:CHANNEL CH1 indicating the error detector channel is set to CH1.

ERRORDetector:CHARacter? (Query Only)

This command queries all character error settings, status, and results.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter?

Examples ERRORDETECTOR:CHARACTER? might return :ERRORDETECTOR:CHARACTER:TEST:STATUS:START STOPPED;LOCK NOT_LOCKED;SIGNAL NO_SIGNAL;MAX_AP MAX_AP_OK; :ERRORDETECTOR:CHARACTER:TEST:TIME:DAYS 0;HOURS 0;MINUTES 3;SECONDS 0, indicating that the test is stopped, not locked, there is no signal, MAX_AP is ok, and the test time is 3 minutes.

ERRORDetector:CHARacter:TEST (No Query Form)

This command initiates and terminates character error testing for the argument START and STOP. Zeroes the character error results for the argument CLEAR. Re-synchronizes the recovered clock when the argument is SYNC.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST <START|STOP|CLEAR|SYNC>

Arguments START initiates character error testing.

STOP terminates character error testing.

CLEAR zeroes the character error count, duration, and error rate.

SYNC re-synchronizes the recovered clock.

Examples ERRORDETECTOR:CHARACTER:TEST START initiates character error testing.

ERRORDetector:CHARacter:TEST:COUNt? (Query Only)

This command queries the number of illegal characters in the signal for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:COUNt?

Examples ERRORDETECTOR:CHARACTER:TEST:COUNT? might return :ERRORDETECTOR:CHARACTER:TEST:COUNT 0, indicating that the number of illegal characters in the signal for character error testing is 0.

ERRORDetector:CHARacter:TEST:DISPCOUNT? (Query Only)

This command queries the number of disparity errors in the signal for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:DISPCOUNT?

Examples ERRORDETECTOR:CHARACTER:TEST:DISPCOUNT? might return :ERRORDETECTOR:CHARACTER:TEST:DISPCOUNT 0, indicating that the number of disparity errors in the signal for character error testing is 0.

ERRORDetector:CHARacter:TEST:DISPRATE? (Query Only)

This command queries the calculated disparity error rate for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:DISPRATE?`

Examples `ERRORDETECTOR:CHARACTER:TEST:DISPRATE?` might return `:ERRORDETECTOR:CHARACTER:TEST:DISPRATE 0.0000`, indicating that the calculated disparity error rate for character error testing is 0.0000.

ERRORDetector:CHARacter:TEST:DURATION? (Query Only)

This command queries the elapsed duration in units of characters tested.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:DURATION?`

Examples `ERRORDETECTOR:CHARACTER:TEST:DURATION?` might return `:ERRORDETECTOR:CHARACTER:TEST:DURATION 2876`, indicating that the elapsed duration in units of characters tested is 2876.

ERRORDetector:CHARacter:TEST:MAXALIGNS? (Query Only)

This command queries the maximum consecutive aligns encountered for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:MAXALIGNS?`

Examples `ERRORDETECTOR:CHARACTER:TEST:MAXALIGNS?` might return `:ERRORDETECTOR:CHARACTER:TEST:MAXALIGNS 2`, indicating that the maximum consecutive aligns encountered for character error testing is 2.

ERRORDetector:CHARacter:TEST:RATE? (Query Only)

This command queries the calculated illegal character error rate for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:RATE?`

Examples `ERRORDETECTOR:CHARACTER:TEST:RATE?` might return `:ERRORDETECTOR:CHARACTER:TEST:RATE 0.00000`, indicating that the calculated illegal character error rate for character error testing is 0.0000.

ERRORDetector:CHARacter:TEST:RESults? (Query Only)

This command queries all of the character error test results.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:RESults?`

Returns `COUNT` is the number of illegal characters in the signal.

`DURATION` is the elapsed duration in units of characters tested.

`RATE` is the calculated illegal character error rate.

`MAXALIGNS` is the maximum consecutive aligns encountered.

`DISPCOUNT` is the number of disparity errors in the signal.

DISPRATE is the calculated disparity error rate.

SECONDS is the elapsed duration time in seconds.

TIME is the elapsed duration time in DD:HH:MM:SS.

Examples ERRORDETECTOR:CHARACTER:TEST:RESULTS? might return :ERRORDETECTOR:CHARACTER:TEST:RESULTS 0,2365,0.0000,2,2,0.0000,16,0,0,2,16.., indicating that the number of illegal characters in the signal is 0, elapsed duration in units of characters tested is 2365, calculated illegal character error rate is 0.0000, maximum consecutive aligns encountered is 2, number of disparity errors in the signal is 2, calculated disparity error rate is 0.0000, elapsed duration time in seconds is 16, and the elapsed duration time is 0 days, 0 hours, 2 minutes, and 16 seconds.

ERRORDetector:CHARacter:TEST:SECOnds? (Query Only)

This command queries the results of the elapsed time (in seconds) for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:SECOnds?

Examples ERRORDETECTOR:CHARACTER:TEST:SECONDS? might return :ERRORDETECTOR:CHARACTER:TEST:SECONDS 30, indicating that elapsed time for character error testing is 30 seconds.

ERRORDetector:CHARacter:TEST:STATUS? (Query Only)

This command queries all the character error test status.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:STATUS?

Returns COUNTING or NOT_COUNTING for START status for character error test.

LOCKED or NOT_LOCKED for LOCK status for character error test.

SIGNAL or NO_SIGNAL for SIGNAL status for character error test.

MAX_AP_OK or MAX_AP_FAIL for MAX_AP status for character error test.

Examples ERRORDETECTOR:CHARACTER:TEST:STATUS? might return :ERRORDETECTOR:CHARACTER:TEST:STATUS STOPPED;LOCKED;NO_SIGNAL;MAX_AP_OK, indicating that the status for character error test are STOPPED, LOCKED, NO_SIGNAL, and MAX_AP_OK.

ERRORDetector:CHARacter:TEST:STATUS:LOCK? (Query Only)

This command queries the LOCK status for the character error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:STATUS:LOCK?

Returns LOCKED or NOT_LOCKED for LOCK status for the character error test.

Examples ERRORDETECTOR:CHARACTER:TEST:STATUS:LOCK? might return :ERRORDETECTOR:CHARACTER:TEST:STATUS:LOCK LOCKED, indicating that the recovered clock reference signal is LOCKED for error test.

ERRORDetector:CHARacter:TEST:STATUS:MAX_AP? (Query Only)

This command queries the MAX_AP status for the character error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:STATUS:MAX_AP?

Returns MAX_AP_OK or MAX_AP_FAIL for MAX_AP status for character error test.

Examples ERRORDETECTOR:CHARACTER:TEST:STATUS:MAX_AP? might return :ERRORDETECTOR:CHARACTER:TEST:STATUS:MAX_AP MAX_AP_OK, indicating that the MAX_AP status is ok for character error test.

ERRORDetector:CHARacter:TEST:STATUS:SIGNAL? (Query Only)

This command queries the SIGNAL status for the character error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:STATUS:SIGNAL?

Returns SIGNAL or NO_SIGNAL for SIGNAL status for character error test.

Examples ERRORDETECTOR:CHARACTER:TEST:STATUS:SIGNAL? might return :ERRORDETECTOR:CHARACTER:TEST:STATUS:SIGNAL SIGNAL, indicating that the signal is connected for character error test.

ERRORDetector:CHARacter:TEST:STATUS:START? (Query Only)

This command queries the START status for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:CHARacter:TEST:STATUS:START?

Returns COUNTING or NOT_COUNTING for START status for character error test.

Examples ERRORDETECTOR:CHARACTER:TEST:STATUS:START? might return :ERRORDETECTOR:CHARACTER:TEST:STATUS:START COUNTING, indicating that the character error comparisons are being made.

ERRORDetector:CHARacter:TEST:TIME? (Query Only)

This command queries the elapsed time (in days, hours, minutes, and seconds) for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:TIME?`

Examples `ERRORDETECTOR:CHARACTER:TEST:TIME?` might return `:ERRORDETECTOR:CHARACTER:TEST:TIME 0,0,2,14`, indicating that the elapsed time is 0 days, 0 hours, 2 hours, and 14 seconds for character error testing.

ERRORDetector:CHARacter:TEST:TIME:DAYS? (Query Only)

This command queries the elapsed time days component for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:TIME:DAYS?`

Examples `ERRORDETECTOR:CHARACTER:TEST:TIME:DAYS?` might return `:ERRORDETECTOR:CHARACTER:TEST:TIME:DAYS 0`, indicating that the elapsed time days component for character error testing is 0.

ERRORDetector:CHARacter:TEST:TIME:HOURS? (Query Only)

This command queries the elapsed time hours component for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:TIME:HOURS?`

Examples `ERRORDETECTOR:CHARACTER:TEST:TIME:HOURS?` might return `:ERRORDETECTOR:CHARACTER:TEST:TIME:HOURS 0`, indicating that the elapsed time hours component for character error testing is 0.

ERRORDetector:CHARacter:TEST:TIME:MINUTES? (Query Only)

This command queries the elapsed time minutes component for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:TIME:MINUTES?`

Examples `ERRORDETECTOR:CHARACTER:TEST:TIME:MINUTES?` might return `:ERRORDETECTOR:CHARACTER:TEST:TIME:MINUTES 0`, indicating that the elapsed time minutes component for character error testing is 0.

ERRORDetector:CHARacter:TEST:TIME:SECOnds? (Query Only)

This command queries the elapsed time seconds component for character error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:CHARacter:TEST:TIME:SECOnds?`

Examples `ERRORDETECTOR:CHARACTER:TEST:TIME:SECONDS?` might return `:ERRORDETECTOR:CHARACTER:TEST:TIME:SECONDS 25`, indicating that the elapsed time seconds component for character error testing is 25.

ERRORDetector:DURATION:COUNt

This command sets or queries the test duration count as the number of bits, frames, symbols, or characters to be tested for error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:DURATION:COUNt <NR1>
ERRORDetector:DURATION:COUNt?
```

Arguments <NR1> indicates the number of bits, frame, symbols, or characters to be tested for the test duration count.

Supports 48bit values. The bit ranges from 0 to 00x7FFFFFFFFFF. A value of –1, which is the default value, indicates count forever.

Examples ERRORDETECTOR:DURATION:COUNT 100000000000 sets the test duration count for error testing to 100000000000.

ERRORDETECTOR:DURATION:COUNT? might return :ERRORDETECTOR:DURATION:COUNT 100000000000, indicates that the test duration count for error testing is set to 100000000000.

ERRORDetector:DURATION:SECOnds

This command sets or queries the test duration in seconds.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:DURATION:SECOnds <NR1>
ERRORDetector:DURATION:SECOnds?
```

Arguments <NR1> is the test duration in seconds.

Examples `ERRORDETECTOR:DURATION:SECONDS 30` sets the test duration to 30 seconds.

`ERRORDETECTOR:DURATION:SECONDS?` might return `:ERRORDETECTOR:DURATION:SECONDS 30`, indicating that the test duration is 30 seconds.

ERRORDetector:DURATION:TIME

This command sets or queries the test duration time in days, hours, minutes, and seconds.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:DURATION:TIME <NR1>
ERRORDetector:DURATION:TIME?
```

Arguments <NR1> is the test duration time in days, hours, minutes, and seconds. It is in the format <DD, HH, MM, SS>

Examples `ERRORDETECTOR:DURATION:TIME 02,12,59,30` sets the test duration time to 2 days, 12 hours, 59 minutes, and 30 seconds.

`ERRORDETECTOR:DURATION:TIME?` might return `:ERRORDETECTOR:DURATION:TIME 02,12,59,30`, indicating that the test duration time is two days, 12 hours, 59 minutes, and 30 seconds.

ERRORDetector:DURATION:TIME:DAYS

This command sets or queries the test duration time days component.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:DURATION:TIME:DAYS <NR1>
ERRORDetector:DURATION:TIME:DAYS?
```

Arguments <NR1> is a number for the test duration time days component.

Examples `ERRORDETECTOR:DURATION:TIME:DAYS 1` sets the test duration time days component for error testing to 1.

`ERRORDETECTOR:DURATION:TIME:DAYS?` might return `:ERRORDETECTOR:DURATION:TIME:DAYS 02`, indicating that the test duration time days component is set to 2.

ERRORDetector:DURATION:TIME:HOURS

This command sets or queries the test duration time hours component.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:DURATION:TIME:HOURS <NR1>
ERRORDetector:DURATION:TIME:HOURS?
```

Arguments `<NR1>` is a number for the test duration time hours component.

Examples `ERRORDETECTOR:DURATION:TIME:HOURS 5` sets the test duration time for error testing to five hours.

`ERRORDETECTOR:DURATION:TIME:HOURS?` might return `:ERRORDETECTOR:DURATION:TIME:HOURS 05`, indicating that the test duration time hours component is set to 5.

ERRORDetector:DURATION:TIME:MINUTES

This command sets or queries the test duration time minutes component for error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:DURATION:TIME:MINUTES <NR1>
ERRORDetector:DURATION:TIME:MINUTES?
```

Arguments `<NR1>` is a number for the test duration time minutes component.

Examples `ERRORDETECTOR:DURATION:TIME:MINUTES 5` sets the test duration time minutes component for error testing to 5.

`ERRORDETECTOR:DURATION:TIME:MINUTES?` might return `:ERRORDETECTOR:DURATION:TIME:MINUTES 05`, indicating that the test duration time minutes component is set to 5.

ERRORDetector:DURATION:TIME:SECOnds

This command sets or queries the test duration time seconds component for error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:DURATION:TIME:SECOnds <NR1>
ERRORDetector:DURATION:TIME:SECOnds?
```

Arguments `<NR1>` is a number for the test duration time seconds component.

Examples `ERRORDETECTOR:DURATION:TIME:SECONDS 25` sets the test duration time component for error testing to 25.

`ERRORDETECTOR:DURATION:TIME:SECONDS?` might return `:ERRORDETECTOR:DURATION:TIME:SECONDS 25`, indicating that the test duration time seconds component is set to 25.

ERRORDetector:ERRORLIMIT

This command sets or queries the error limit value to use when STOPWHEN is ERROR.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:ERRORLIMIT <NR1>
ERRORDetector:ERRORLIMIT?
```

Arguments <NR1> is the maximum number of errors.

Examples ERRORDETECTOR:ERRORLIMIT 2 sets the error limit value to 2.

ERRORDETECTOR:ERRORLIMIT? might return :ERRORDETECTOR:ERRORLIMIT 1, indicating that the error limit value is 1.

ERRORDetector:FILE:RECAll (No Query Form)

This command initiates a file recall of error detector parameters from a text file. Argument is the file name.

The setup files are supplied with the instrument. You can also create your own setup files. The setup for the USB CP0_SKP signal test pattern is supplied in "C:\Users\Public\Tektronix\TekScope\ErrorDetector\UsbCP0_SKPsymbolErrorSetup.txt".

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:FILE:RECAll <fileName>

Arguments <fileName> is the file name and file location. The default location is "C:\Users\Public\Tektronix\TekScope\ErrorDetector" and the setup file is in TXT format.

Examples ERRORDETECTOR:FILE:RECALL "C:\Users\Public\Tektronix\TekScope\ErrorDetector\SataFrameComplianceFrameErrorSetup.txt" recalls the error detector parameters from the setup file "SataFrameComplianceFrameErrorSetup.txt".

ERRORDetector:FILE:SAVe (No Query Form)

This command initiates a file save of error detector parameters to a text file. Argument is the file name.

The setup files are supplied with the instrument. You can also save your own setup files. For example, the setup for the USB CP0_SKP signal test pattern is supplied in "C:\Users\Public\Tektronix\TekScope\ErrorDetector\UsbCP0_SKPsymbolErrorSetup.txt".

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FILE:SAVe <fileName>`

Arguments `<fileName>` is the file name and file location. The default location is "C:\Users\Public\Tektronix\TekScope\ErrorDetector" and the setup file is in TXT format.

Examples `ERRORDETECTOR:FILE:SAVE "C:\Users\Public\Tektronix\TekScope\ErrorDetector\MySataFrameComplianceFrameErrorSetup.txt"` saves the error detector parameters to the setup file "`MySataFrameComplianceFrameErrorSetup.txt`".

ERRORDetector:FONTSIze

Sets or queries error detector font size selection. Currently, the font size only applied the error detector UI control window.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:FONTSIze {DEFAULT | LARGE | XLARGE
ERRORDetector:FONTSIze?
```

Arguments `DEFAULT` sets the font size to the default size.

`LARGE` sets the font size to large.

`XLARGE` sets the font size to extra large.

Examples `ERRORDETECTOR:FONTSIZE XLARGE` sets the error detector font size to extra large.

`ERRORDETECTOR:FONTSIZE?` might return :ERRORDETECTOR:FONTSIZE LARGE indicating the error detector is using the large font size.

ERRORDetector:FRAME? (Query Only)

This command queries all frame error settings, status, and results.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAME?`

Examples `ERRORDETECTOR:FRAME?` might return `:ERRORDETECTOR:FRAME:SOF "00111110101011011110110011101100";EOF "00111110101011011010101110101011";INITIALCRCVALUE 1379029042;TEST:STATUS:START STOPPED;LOCK NOT_LOCKED;SIGNAL NO_SIGNAL;MAX_AP MAX_AP_OK`, indicating the start of frame is 00111110101011011110110011101100, the end of frame is 00111110101011011010101110101011, the initial CRC value is 1379029042, the test is stopped, not locked, there is no signal, MAX_AP is ok.

ERRORDetector:FRAme:EOF

This command sets or queries the End of Frame for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:FRAme:EOF <Qstring>
ERRORDetector:FRAme:EOF?
```

Arguments `<Qstring>` is a quoted string representing a 32-bit pattern.

Examples `ERRORDETECTOR:FRAME:EOF “00111110101011011010101110101011”` sets the end of frame to "00111110101011011010101110101011".

`ERRORDETECTOR:FRAME:EOF?` might return `ERRORDETECTOR:FRAME:EOF “00111110101011011010101110101011”`, indicating that the end of frame is “00111110101011011010101110101011”.

ERRORDetector:FRAme:INITIALCRCVALue

This command sets or queries the initial CRC value for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:FRAme:INITIALCRCVALue <NR1>
ERRORDetector:FRAme:INITIALCRCVALue?
```

Arguments <NR1> is a value defined by the selected standard.

Examples ERRORDETECTOR:FRAME:INITIALCRCVALUE 1379029042 sets the CRC value to 1379029042.

ERRORDETECTOR:FRAME:INITIALCRCVALUE? might return :ERRORDETECTOR:FRAME:INITIALCRCVALUE 1379029042, indicating that the initial CRC value is set to 1379029042, the SATA default.

ERRORDetector:FRAme:SOF

This command sets or queries the Start of Frame for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:FRAme:SOF <Qstring>
ERRORDetector:FRAme:SOF?
```

Arguments <Qstring> is a quoted string representing a 32-bit pattern.

Examples ERRORDETECTOR:FRAME:SOF "00111110101011011110110011101100" sets the start of frame.

ERRORDETECTOR:FRAME:SOF? might return :ERRORDETECTOR:FRAME:SOF "00111110101011011110110011101100", indicating that the start of frame is set to "00111110101011011010101110101011".

ERRORDetector:FRAme:TEST

This command and query initiates and terminates frame error testing for the arguments START and STOP. Zeroes the frame error results for the argument CLEAR. Re-synchronizes the recovered clock for the argument SYNC.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:FRAme:TEST <START|STOP|CLEAR|SYNC>
ERRORDetector:FRAme:TEST ?
```

Arguments

`START` initiates the frame error test counting of errors and duration.

`STOP` terminates the frame error test counting of frame errors and duration.

`CLEAR` zeroes the frame error test count, duration, and rate.

`SYNC` re-synchronizes the recovered clock.

Examples

`ERRORDETECTOR:FRAME:TEST START` initiates the frame error test counting.

`ERRORDETECTOR:FRAME:TEST?` might return `:ERRORDETECTOR:FRAME:TEST STOPPED;LOCKED;SIGNAL;MAX_AP_OK;0;0;0;14`, indicating that the signal is stopped and locked for frame error testing. The status of MAX_AP is ok and the elapsed duration is 14 seconds.

ERRORDetector:FRAme:TEST:BADCHARS? (Query Only)

This command queries the illegal character count for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:FRAme:TEST:BADCHARS?
```

Examples

`ERRORDETECTOR:FRAME:TEST:BADCHARS?` might return `:ERRORDETECTOR:FRAME:TEST:BADCHARS 0`, indicating that the illegal character count for the frame error test is 0.

ERRORDetector:FRAme:TEST:COUNt? (Query Only)

This command queries the test error count for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:COUNt?`

Examples `ERRORDETECTOR:FRAME:TEST:COUNT?` might return `:ERRORDETECTOR:FRAME:TEST:COUNT 0`, indicating that the test error count for the frame error testing is 0.

ERRORDetector:FRAme:TEST:DISParity? (Query Only)

This command queries the disparity error count for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:DISParity?`

Examples `ERRORDETECTOR:FRAME:TEST:DISPARITY?` might return `:ERRORDETECTOR:FRAME:TEST:DISPARITY 0`, indicating that the disparity error count for the frame error test is 0.

ERRORDetector:FRAme:TEST:DURATION? (Query Only)

This command queries the elapsed duration in number of frames.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:DURATION?`

Examples `ERRORDETECTOR:FRAME:TEST:DURATION?` might return `:ERRORDETECTOR:FRAME:TEST:DURATION 4253659`, indicating that the elapsed duration is 4253659 frames.

ERRORDetector:FRAme:TEST:MAXALIGNS? (Query Only)

This command queries the result of the maximum consecutive aligns encountered for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:MAXALIGNS?`

Examples `ERRORDETECTOR:FRAME:TEST:MAXALIGNS?` might return `:ERRORDETECTOR:FRAME:TEST:MAXALIGNS 4`, indicating that the maximum consecutive aligns encountered for frame error test is 4.

ERRORDetector:FRAme:TEST:RATE? (Query Only)

This command queries the calculated frame error rate.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:RATE?`

Examples `ERRORDETECTOR:FRAME:TEST:RATE?` might return `:ERRORDETECTOR:FRAME:TEST:RATE 0.0000`, indicating that the calculated frame error rate is 0.0000.

ERRORDetector:FRAme:TEST:RESults? (Query Only)

This command queries all the results for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:RESults?`

Returns `COUNT` is the frame error count (number of bad frames).

`DURATION` is the elapsed duration in units of frames tested.

`RATE` is the calculated frame error rate.

`MAXALIGNS` is the maximum consecutive aligns encountered.

`BADCHARS` is the illegal character count.

`DISPARITY` is the disparity error count.

`SECONDS` is the elapsed duration time in seconds.

`TIME` is the elapsed duration time in DD:HH:MM:SS.

Examples `ERRORDETECTOR:FRAME:TEST:RESULTS?` might return `:ERRORDETECTOR:FRAME:TEST:RESULTS 0,4253659,0.0000,4,0,0,171,0,0,2,51`, indicating that the error count is four, error duration is 4253659, error rate is 0.0000, error maximum consecutive aligns is 4, illegal character count is 0, disparity error count is 0, elapsed duration time is 171 seconds, and elapsed duration time is 0 days, 0 hours, 2 minutes, and 51 seconds.

ERRORDetector:FRAme:TEST:SECOnds? (Query Only)

This command queries the result of elapsed duration in seconds for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:SECOnds?`

Examples `ERRORDETECTOR:FRAME:TEST:SECONDS?` might return `:ERRORDETECTOR:FRAME:TEST:SECONDS 20`, indicating that the elapsed duration is 20 seconds for frame error testing.

ERRORDetector:FRAme:TEST:STATUS? (Query Only)

This command queries all of the status for frame error status.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:STATUS?`

Returns `COUNTING` or `NOT_COUNTING` for `START` status for frame error test.

`LOCKED` or `NOT_LOCKED` for `LOCK` status for frame error test.

`SIGNAL` or `NO_SIGNAL` for `SIGNAL` status for frame error test.

`MAX_AP_OK` or `MAX_AP_FAIL` for `MAX_AP` status for frame error test.

Examples `ERRORDETECTOR:FRAME:TEST:STATUS?` might return `:ERRORDETECTOR:FRAME:TEST:STATUS STOPPED;LOCKED;NO_SIGNAL;MAX_AP_OK`, indicating that the frame error status is STOPPED, LOCKED, NO_SIGNAL, and MAX_AP_OK.

ERRORDetector:FRAme:TEST:STATUS:LOCK? (Query Only)

This command queries the LOCK status for the frame error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:STATUS:LOCK?`

Returns `LOCKED` or `NOT_LOCKED` for `LOCK` status for frame error test.

Examples `ERRORDETECTOR:FRAME:TEST:STATUS:LOCK?` might return `:ERRORDETECTOR:FRAME:TEST:STATUS:LOCK LOCKED`, indicating that the frame is LOCKED for the error test.

ERRORDetector:FRAme:TEST:STATUS:MAX_AP? (Query Only)

This command queries the MAX_AP status for the frame error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:STATUS:MAX_AP?`

Returns `MAX_AP_OK` or `MAX_AP_FAIL` for `MAX_AP` status for frame error test.

Examples `ERRORDETECTOR:FRAME:TEST:STATUS:MAX_AP?` might return `:ERRORDETECTOR:FRAME:TEST:STATUS:MAX_AP MAX_AP_OK`, indicating that the MAX_AP status is ok for the frame error test.

ERRORDetector:FRAme:TEST:STATUS:SIGNAL? (Query Only)

This command queries the SIGNAL status for the frame error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:STATUS:SIGNAL?`

Returns `SIGNAL` or `NO_SIGNAL` for `SIGNAL` status for frame error test.

Examples `ERRORDETECTOR:FRAME:TEST:STATUS:SIGNAL?` might return `:ERRORDETECTOR:FRAME:TEST:STATUS:SIGNAL SIGNAL`, indicating that signal is connected for the frame error test. NO_SIGNAL would indicate the lack of the signal.

ERRORDetector:FRAme:TEST:STATUS:START? (Query Only)

This command returns the START status for frame error tests.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:STATUS:START?`

Returns `COUNTING` or `NOT_COUNTING` for `START` status for frame error test.

Examples `ERRORDETECTOR:FRAME:TEST:STATUS:START?` might return `:ERRORDETECTOR:FRAME:TEST:STATUS:START COUNTING`, indicating that the START status is COUNTING.

ERRORDetector:FRAme:TEST:TIME? (Query Only)

This command queries the elapsed time (in days, hours, minutes, and seconds) for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:TIME?`

Examples `ERRORDETECTOR:FRAME:TEST:TIME?` might return `:ERRORDETECTOR:FRAME:TEST:TIME 0,0,0,14`, indicating that the elapsed time is 0 days, 0 hours, 0 minutes, and 14 seconds for frame error testing.

ERRORDetector:FRAme:TEST:TIME:DAYS? (Query Only)

This command queries the elapsed time days component for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:TIME:DAYS?`

Examples `ERRORDETECTOR:FRAME:TEST:TIME:DAYS?` might return `:ERRORDETECTOR:FRAME:TEST:TIME:DAYS 0`, indicating that the elapsed time days component is 0 for frame error testing.

ERRORDetector:FRAme:TEST:TIME:HOURS? (Query Only)

This command queries the elapsed time hours component for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:TIME:HOURS?`

Examples `ERRORDETECTOR:FRAME:TEST:TIME:HOURS?` might return `:ERRORDETECTOR:FRAME:TEST:TIME:HOURS 0`, indicating that the elapsed time hours component is 0 for frame error testing.

ERRORDetector:FRAme:TEST:TIME:MINUTES? (Query Only)

This command queries the elapsed time minutes component for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:TIME:MINUTES?`

Examples `ERRORDETECTOR:FRAME:TEST:TIME:MINUTES?` might return `:ERRORDETECTOR:FRAME:TEST:TIME:MINUTES 0`, indicating that the elapsed time minutes component is 0 for frame error testing.

ERRORDetector:FRAme:TEST:TIME:SECOnds? (Query Only)

This command queries the elapsed time seconds component for frame error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:FRAme:TEST:TIME:SECOnds?`

Examples `ERRORDETECTOR:FRAME:TEST:TIME:SECONDS?` might return `:ERRORDETECTOR:FRAME:TEST:TIME:SECONDS 25`, indicating that the elapsed time seconds component is 25 for frame error testing.

ERRORDetector:MAXALIGNS

This command sets or queries the maximum consecutive align primitives before a MAX_AP_FAIL error is reported.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:MAXALIGNS <NR1>
ERRORDetector:MAXALIGNS?
```

Arguments `<NR1>` is a integer. The limit values are 0 to 63 and the default is 8.

Examples `ERRORDETECTOR:MAXALIGNS 6` sets the maximum consecutive align primitives to 6 before an error occurs. Exceeding the maximum limit sets the MAX_AP_FAIL status error.

`ERRORDETECTOR:MAXALIGNS?` might return `:ERRORDETECTOR:MAXALIGNS 11`, indicating that the maximum consecutive align primitives is 11 before an error occurs. Exceeding the maximum limit sets the MAX_AP_FAIL status error.

ERRORDetector:PATTERNNAME

This command sets or queries the pattern name stored in the setup file. Setting this name has no functional effect on the instrument, but is a handy reminder to humans as to which setup is in effect.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:PATTERNNAME <Qstring>
ERRORDetector:PATTERNNAME?
```

Arguments

<Qstring> is a quoted string representing a pattern name.

Examples

ERRORDETECTOR:PATTERNNAME "FrameComp" sets the pattern name to FrameComp.

ERRORDETECTOR:PATTERNNAME? might return :ERRORDETECTOR:PATTERNNAME "FrameComp", indicating that the pattern name is FrameComp. When a setup file is saved or recalled, the pattern name is displayed on the user interface or program interface.

ERRORDetector:PREset

This command sets or queries error detector font preset selection. A number of preset setups are selected by this parameter to cover the more common cases. The preset names attempt to indicate the standard, signal pattern, and test type employed. The bit rate appropriate for the standard is used. The text files containing the preset setups are located in the C:\Users\Public\Tektronix\TekScope\ErrorDetector directory on WIN7.

You may select CUSTOM as a preset value, and save or recall your own custom setups. You may want to recall one of the standard preset setups, modify some of the parameters, and then save it as a custom setup for recall at a later time. This same behavior is supported on the error detector User Interface. The SATA3_FRAME preset expects the SATA3 Compliance Pattern. USB3_SYMBOL preset expects the USB3 standard CP0_SKP signal. You can set a PATTERNNAME for each setup.

NOTE. *Not available on some models.*

Group

Error Detector

Syntax

```
ERRORDetector:PREset {SATA1_CJTPAT_BIT | SATA2_CJTPAT_BIT |
SATA3_CJTPAT_BIT | SATA3_FRAME | SATA3_CHAR | SATA3_HFTP_BIT
| SATA3_LBP_BIT | SATA3_LFTP_BIT | SATA3_MFTP_BIT |
USB3_SYMBOL | USB3_CHAR | PCIE1_COMP_BIT | PCIE2_COMP_BIT |
ANY_CJTPAT_BIT | ANY_CJTPAT_CHAR | CUSTOM}
```

Arguments

Arguments are the file names of preset setups.

Examples `ERRORDETECTOR:PRESET USB3_CHAR` uses the USB3_CHAR preset.

`ERRORDETECTOR:PRESET?` might return :ERRORDETECTOR:PRESET SATA3_CJTPAT_BIT indicating the error detector is using the SATA3_CJTPAT_BIT presets.

ERRORDetector:PREset:APPLY (No Query Form)

This command causes selected preset setup to be applied. Until this command is received by the instrument, the selected preset has not been applied. This mimics the user interface operation, which allows window shopping various preset setups, without actually applying them to the instrument setup.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:PREset:APPLY
```

Examples `ERRORDETECTOR:PRESET:APPLY` applies the selected preset.

ERRORDetector:SAVEIMAGE

Sets or queries error detector Save Image control. When set to ON, a screen shot will be made when the error detector detects an error (because detecting an error triggers the scope).

The images are saved in the C:\Users\<yourName>\Tektronix\TekScope\SaveOnTrigger directory. A default limit of 10 screen shots prevents overflowing your disk drive should the error detector sense massive errors, such as when you disconnect the signal. If you also set the SendEmail parameter to ON, the saved image (screen shot) will be emailed to the recipient (set elsewhere in the trigger PI). SaveImage is an alternate way of setting the Save on Trigger actions defined elsewhere in the PI.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:SAVEIMAGE {OFF | ON}
ERRORDetector:SAVEIMAGE?
```

Arguments OFF turns off the save image feature.

ON turns on the save image feature.

Examples `ERRORDETECTOR:SAVEIMAGE ON` turns the save image feature on.

`ERRORDETECTOR:SAVEIMAGE?` might return :ERRORDETECTOR:SAVEIMAGE OFF indicating that the error detector save image feature is off.

ERRORDetector:SAVEWFM

This command sets or queries error detector Save Waveform (WFM) control. When set to ON, a waveform object will be made when the error detector detects an error (because detecting an error triggers the instrument).

The waveforms are saved in the C:\Users\<yourName>\Tektronix\TekScope\SaveOnTrigger directory. A default limit of 10 screen shots prevents overflowing your disk drive should the error detector sense massive errors, such as when you disconnect the signal.

If you also set the SendEmail parameter to ON, the saved waveform (wfm object) is emailed to the recipient (set elsewhere in the trigger PI) as an attachedment. SaveImage is an alternate way of setting the Save on Trigger actions defined elsewhere in the PI.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:SAVEWFM {OFF | ON}
ERRORDetector:SAVEWFM?
```

Arguments `OFF` turns off the error detector save waveform feature.

`ON` turns on the error detector save waveform feature.

Examples `ERRORDETECTOR:SAVEWFM ON` turns on the error detector save waveform feature.

`ERRORDETECTOR:SAVEWFM?` might return :ERRORDETECTOR:SAVEWFM OFF indicating that the error detector save waveform feature is off.

ERRORDetector:SCRAMBLED

This command sets or queries the status of the error detection data scrambling option.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:SCRAMBLED {ON|OFF}
ERRORDetector:SCRAMBLED?
```

Arguments `ON` enables the error detection data scrambling option. This is the default option.

`OFF` disables the error detection data scrambling option.

Examples `ERRORDETECTOR:SCRAMBLED ON` specifies that the error detection data scrambling option is enabled.

`ERRORDETECTOR:SCRAMBLED?` might return `:TRIGGER:A:SERIAL:ERRORDETECTOR:SCRAMBLED ON`, indicating that the error detection data scrambling option is enabled.

ERRORDetector:SENDEMAIL

This command sets or queries error detector Send Email control. When set to ON, a email will be sent to the recipient, defined elsewhere in the PI, when the error detector detects an error (because detecting an error triggers the instrument). The default number of emails sent is 1, so that you do not overflow your inbox.

If you also set the SaveImage or SaveWfm parameters to ON, the email will contain these items. Send Email is an alternate way of setting the E-mail on Trigger actions defined elsewhere in the PI.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:SENDEMAIL {OFF | ON}
ERRORDetector:SENDEMAIL?
```

Arguments `OFF` disables the send email feature.

`ON` enables the send email feature.

Examples `ERRORDETECTOR:SENDEMAIL` ON enables the error detector send mail feature

`ERRORDETECTOR:SENDEMAIL?` might return :ERRORDETECTOR:SENDEMAIL OFF indicating that the error detector send email feature is off.

ERRORDetector:SIGnaltype

This command sets or queries error detector Signal Type control. Setting the signal type establishes the bit rate appropriate for the standard, as well as establishing the testing algorithm. Custom bit rates may be used as well. See the ERRORDetector:BITRATE and ERRORDetector:BITRATE:VALue commands.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:SIGnaltype {SATAGEN1 | SATAGEN2 | SATAGEN3 |
USB3 | PCIEGEN1 | PCIEGEN2 | ANY8B10B | PRBS7 | PRBS9}
ERRORDetector:SIGnaltype?
```

Arguments Sets the error detector signal type to the specified signal.

Examples `ERRORDETECTOR:SIGNALTYPE USB3` sets the signal type to USB3.

`ERRORDETECTOR:SIGNALTYPE?` might return :ERRORDETECTOR:SIGNALTYPE SATAGEN3 indicating that the error detector signal type is set to SATAGEN3.

ERRORDetector:SKIPSETPRIMitive:MINus<x>? (Query Only)

Queries the RD- skip order set array of bit string values. The PI array elements may be accessed individual via MINUS1, MINUS2, MINUS3.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SKIPSETPRIMitive:MINus<x>?`

Examples `ERRORDetector:SKIPSETPRIMitive:MINus?` might return `:ERRORDETECTOR:SKIPSETPRIMITIVE:MINUS “0011111010”`.

ERRORDetector:SKIPSETPRIMitive:PLUS<x>? (Query Only)

Queries the skip order RD+ array of bit string values. The PI array elements may be accessed individual via PLUS1, PLUS2, PLUS3, and PLUS4.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SKIPSETPRIMitive:PLUS<x>?`

Examples `ERRORDetector:SKIPSETPRIMitive:MINus?` might return `:ERRORDETECTOR:SKIPSETPRIMITIVE:PLUS, “1100000101”` indicating the first RD+ skip order bit string element is "1100000101”.

ERRORDetector:SKIPSETPRIMitive:STATE

This command sets or queries the state (ON/OFF) of the skip order set primitive.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:SKIPSETPRIMitive:STATE {ON|OFF|<NR1>}
ERRORDetector:SKIPSETPRIMitive:STATE?
```

Arguments `ON` enables the skip order set primitive.

`OFF` disables the skip order set primitive.

`<NR1>` = 0 disables the skip order set primitive; any other value enables the skip order set primitive.

Examples `ERRORDETECTOR:SKIPSETPRIMITIVE:STATE ON`, enables the skip order set primitive.

`ERRORDETECTOR:SKIPSETPRIMITIVE:STATE?` might return `:ERRORDETECTOR:SKIPSETPRIMITIVE:STATE ON`, indicating that the skip order set option is enabled.

ERRORDetector:SKIPSETPRIMitive:SYMBOLS

Sets or queries the skip order set values in symbolic form. You can set one or more symbols with the command, but they must be done in order. The query returns all four symbols. A symbol may be set or queried as "X" to indicate "don't care."

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:SKIPSETPRIMitive:SYMBOLS <Qstring> {<Qstring>}
ERRORDetector:SKIPSETPRIMitive:SYMBOLS?
```

Arguments `<Qstring>` is a quoted strings representing the symbol, such as "K28.5".

Examples `ERRORDETECTOR:SKIPSETPRIMITIVE:SYMBOLS "K28.5".` sets the first skip order set symbol to K28.5.

`ERRORDETECTOR:SKIPSETPRIMITIVE:SYMBOLS "K28.5","D10.2".` sets the first and second skip order set symbols to K28.5 and D10.2, respectively.

`ERRORDETECTOR:SKIPSETPRIMITIVE:SYMBOLS?` might return `:ERRORDETECTOR:SKIPSETPRIMITIVE:SYMBOLS "K28.5","K28.5","K28.5","D10.2".` indicating the four symbols comprising the skip order set.

ERRORDetector:SKIPSETPRIMitive:SYMBOL<x>

Sets or queries the skip order set symbolic characters. The PI array syntax may be used to access the individual elements (SYMBOL1, SYMBOL2, etc.). For example, a skip set symbolic value may be "K28.5," or "X" to indicate "don't care."

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:SKIPSETPRIMitive:SYMBOL<x> <Qstring>
{<Qstring>}
ERRORDetector:SKIPSETPRIMitive:SYMBOL<x>?
```

Arguments

<Qstring> is a quoted string representing the skip set primitive symbol, for example “K28.5”.

Examples

ERRORDetector:SKIPSETPRIMitive:SYMBOL "K28.5" sets the first skip order set symbol to K28.5.

ERRORDetector:SKIPSETPRIMitive:SYMBOL1 "K28.5" also sets the first skip order set symbol to K28.5.

ERRORDetector:SKIPSETPRIMitive:SYMBOL2 "D10.2" sets the second skip order set symbol to D10.2.

ERRORDetector:SKIPSETPRIMitive:SYMBOL? might return :ERRORDETECTOR:SKIPSETPRIMITIVE:SYMBOL “K28.5”. indicating the first symbol comprising the skip order set.

ERRORDetector:SSC

This command sets or queries the status of the spread spectrum clock tracking option.

NOTE. *Not available on some models.*

Group

Error Detector

Syntax

```
ERRORDetector:SSC {ON|OFF}
ERRORDetector:SSC?
```

Arguments

ON enables spread spectrum clock tracking. For error detector, the spread spectrum clock tracking should always be turned on.

OFF disables spread spectrum clock tracking. For serial trigger, the spread spectrum clock tracking is turned off.

Examples

ERRORDETECTOR:SSC ON enables spread spectrum clock tracking option.

ERRORDETECTOR:SSC? might return :ERRORDETECTOR:SSC ON, indicating that spread spectrum clock tracking is enabled.

ERRORDetector:STANdard

This command sets or queries the standard selection for error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:STANdard <LIST>
ERRORDetector:STANdard?
```

Arguments <LIST> is the supported standard. The supported standards are SATA, USB, PCIE, ANY8B10B, PRBS7, and PRBS9.

Examples ERRORDETECTOR:STANDARD SATA sets the standard to SATA.

ERRORDETECTOR:STANDARD? might return :ERRORDETECTOR:STANDARD SATA, indicating that the selected standard is SATA.

ERRORDetector:STATE

This command sets or queries the status of the error option. STATE must be ON to use the error detector feature.

Group Error Detector

Syntax

```
ERRORDetector:STATE {<NR1>|OFF|ON}
ERRORDetector:STATE?
```

Arguments ON enables the software error detector feature.

OFF disables the software error detector feature. This is the default.

<NR1> = 0 disables the error detector; any other value enables the error detector.

Examples ERRORDETECTOR:STATE ON enables the error detector option.

ERRORDETECTOR:STATE? might return :ERRORDETECTOR:STATE OFF, indicating that the error detector option is disabled.

ERRORDetector:STATus? (Query Only)

Queries only the “most significant” or “summary” status of the error detector. All of the status flags for each test type may be obtained from the ERRORdetector:<TESTTYPE>:TEST:STATUS commands.

LOCK refers to the recovered clock. Signal refers to the cable carrying the signal to the scope. SYNC refers to bit error tests that require a sync pattern. STOPPED/COUNTING refer to whether the error detector is testing for errors. MAX_AP refers to whether the error detector has detected the maximum consecutive Align (or SkipSets) Primitives as specified in the ERRORDetector:<TESTTYPE>MAXALIGNS command

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:STATus?
```

Returns LOCKED | NOT_LOCKED | SIGNAL | NO_SIGNAL | SYNC | NOT_SYNC | STOPPED | COUNTING | MAX_AP_OK | MAX_AP_FAIL

Examples `ERRORDETECTOR:STATUS?` might return :ERRORDETECTOR:STATUS NOT_SYNC indicating the status of the error detector is not synced.

ERRORDetector:STOPWHEN

This command sets or queries the stopping condition. The test can be stopped when the count, time, or number of errors elapses. If the STOPWHEN value is MANUAL, the test runs until a TEST STOP command is received.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:STOPWHEN <MANUAL|COUNT|TIME|ERROR>
ERRORDetector:STOPWHEN?
```

Arguments `MANUAL` indicates that the test must be stopped by issuing a TEST STOP command. This is the default.

COUNT stops the test when DURATION:COUNT comparisons are made. The comparisons can be bit, frame, symbol, or character as appropriate for the TEST:TYPE.

TIME stops the test when DURATION:TIME elapses.

ERROR stops the test when the number of errors ≥ ERRORLIMIT.

Examples ERRORDETECTOR:STOPWHEN MANUAL requires the stop test action to stop the test.

ERRORDETECTOR:STOPWHEN? might return :ERRORDETECTOR:STOPWHEN MANUAL, indicating that the stop test action must be used to stop the test.

ERRORDetector:SYMBOL? (Query Only)

This command queries all symbol error settings, status, and results.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:SYMBOL?

Examples ERRORDETECTOR:SYMBOL? might return :ERRORDETECTOR:SYMBOL:TEST:STATUS:START STOPPED;LOCK NOT_LOCKED;SIGNAL NO_SIGNAL;MAX_AP MAX_AP_OK; :ERRORDETECTOR:SYMBOL:TEST:TIME:DAYS 0;HOURS 0;MINUTES 3;SECONDS 0, that the test is stopped, not locked, there is no signal, MAX_AP is ok, and the test time is 3 minutes.

ERRORDetector:SYMBOL:TEST

This command initiates and terminates symbol error testing for the arguments START and STOP. Zeroes the symbol error results for the argument CLEAR. Re-synchronizes the recovered clock for the argument SYNC. This command also queries all of the symbol test settings and results for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:SYMBOL:TEST <START|STOP|CLEAR>
ERRORDetector:SYMBOL:TEST?
```

Arguments

START initiates symbol and bit error testing.

STOP terminates symbol and bit error testing.

CLEAR zeroes the symbol and bit error counts, duration, bit error rate, and symbol error rate.

SYNC re-synchronizes the recovered clock.

Examples

ERRORDETECTOR:SYMBOL:TEST START initiates symbol and bit error testing.

ERRORDETECTOR:SYMBOL:TEST? might return :ERRORDETECTOR:SYMBOL:TEST STOPPED;LOCKED;SIGNAL;MAX_AP_OK;0,0,0,14, indicating that the signal is stopped and locked for symbol and bit error testing. The status of MAX_AP is ok and the elapsed duration is 0 days, 0 hours, 0 minutes, and 14 seconds.

ERRORDetector:SYMBOL:TEST:BADCHARS? (Query Only)

This command queries the illegal character count for symbol error testing.

NOTE. *Not available on some models.*

Group

Error Detector

Syntax

```
ERRORDetector:SYMBOL:TEST:BADCHARS?
```

Examples

ERRORDETECTOR:SYMBOL:TEST:BADCHARS? might return :ERRORDETECTOR:SYMBOL:TEST:BADCHARS 0, indicating that the illegal character count for the symbol error test is 0.

ERRORDetector:SYMBOL:TEST:BITCOUNT? (Query Only)

This command queries the bit error count (number of bad bits) for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:BITCOUNT?`

Examples `ERRORDETECTOR:SYMBOL:TEST:BITCOUNT?` might return `:ERRORDETECTOR:SYMBOL:TEST:BITCOUNT 0`, indicating that the bit error count for the symbol error test is 0.

ERRORDetector:SYMBOL:TEST:BITDURATION? (Query Only)

This command queries the elapsed duration in units of bits tested for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:BITDURATION?`

Examples `ERRORDETECTOR:SYMBOL:TEST:BITDURATION?` might return `:ERRORDETECTOR:SYMBOL:TEST:BITDURATION 24`, indicating that the elapsed duration tested for the symbol error test is 24 bits.

ERRORDetector:SYMBOL:TEST:BITRate? (Query Only)

This command queries the calculated bit error rate for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:BITRate?`

Examples `ERRORDETECTOR:SYMBOL:TEST:BITRATE?` might return `:ERRORDETECTOR:SYMBOL:TEST:BITRATE 0.0000`, indicating that the calculated bit error rate for symbol error testing is 0.0000.

ERRORDetector:SYMBOL:TEST:COUNt? (Query Only)

This command queries the symbol error count for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:COUNt?`

Examples `ERRORDETECTOR:SYMBOL:TEST:COUNT?` might return `:ERRORDETECTOR:SYMBOL:TEST:COUNT 0`, indicating that the symbol error count is 0.

ERRORDetector:SYMBOL:TEST:DISParity? (Query Only)

This command queries the disparity error count for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:DISParity?`

Examples `ERRORDETECTOR:SYMBOL:TEST:DISPARITY?` might return `:ERRORDETECTOR:SYMBOL:TEST:DISPARITY 0`, indicating that the disparity error count for the symbol error test is 0.

ERRORDetector:SYMBOL:TEST:DURATION? (Query Only)

This command queries the elapsed duration (in units of symbols) for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:DURATION?`

Examples `ERRORDETECTOR:SYMBOL:TEST:DURATION?` might return `:ERRORDETECTOR:SYMBOL:TEST:DURATION 12634`, indicating that the elapsed duration in units of symbols for the symbol error test is 12634.

ERRORDetector:SYMBOL:TEST:MAXALIGNS? (Query Only)

This command queries the maximum consecutive skip order sets encountered for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:MAXALIGNS?`

Examples `ERRORDETECTOR:SYMBOL:TEST:MAXALIGNS?` might return `:ERRORDETECTOR:SYMBOL:TEST:MAXALIGNS 3`, indicating that the maximum consecutive skip order sets encountered for the symbol error test is 3.

ERRORDetector:SYMBOL:TEST:RATE? (Query Only)

This command queries the calculated symbol error rate for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:RATE?`

Examples `ERRORDETECTOR:SYMBOL:TEST:RATE?` might return `:ERRORDETECTOR:SYMBol:TEST:RATE 0.0000`, indicating that the calculated symbol error rate tested is 0.0000.

ERRORDetector:SYMBOL:TEST:RESults? (Query Only)

This command queries all of the results for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:RESults?`

Returns `COUNT` is the symbol error count (number of bad 10-bit symbols).

`DURATION` is the elapsed duration in units of symbols tested.

`RATE` is the calculated symbol error rate.

`MAXALIGNS` is the maximum consecutive skipsets encountered.

`BADCHARS` is the illegal character count.

`DISPARITY` is the disparity error count.

`BITCOUNT` is the USB bit error count (number of bad bits).

`BITDURATION` is the elapsed duration in units of bits tested.

`BITRATE` is the calculated USB bit error rate.

`SECONDS` is the elapsed duration time in seconds.

`TIME` is the elapsed duration time in DD:HH:MM:SS.

Examples `ERRORDETECTOR:SYMBOL:TEST:RESULTS?` might return `:ERRORDETECTOR:SYMBOL:TEST:RESULTS 0, 12657, 0.0000, 3, 0, 0, 0, 0.0000, 0,0,1,20`, indicating that the error count is 0, elapsed duration is 12657, error rate is 0.0000, maximum consecutive skipsets encountered is 3, illegal character count is 0, disparity error count is 0, USB bit error count is 0, elapsed duration is 36, calculated USB bit error rate is 0.0000, elapsed duration time is 10 seconds, and elapsed duration time is 0 days, 0 hours, 10 minutes, and 20 seconds.

ERRORDetector:SYMBOL:TEST:SECOnds? (Query Only)

This command queries the elapsed duration time (in seconds) for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:SECOnds?`

Examples `ERRORDETECTOR:SYMBOL:TEST:SECONDS?` might return `:ERRORDETECTOR:SYMBOL:TEST:SECONDS 45`, indicating that the elapsed duration time for symbol error testing is 45 seconds.

ERRORDetector:SYMBOL:TEST:STATUS? (Query Only)

This command queries all of the status for the symbol error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:STATUS?`

Returns `COUNTING` or `NOT_COUNTING` for `START` status for character error test.

`LOCKED` or `NOT_LOCKED` for `LOCK` status for character error test.

`SIGNAL` or `NO_SIGNAL` for `SIGNAL` status for character error test.

`MAX_AP_OK` or `MAX_AP_FAIL` for `MAX_AP` status for character error test.

Examples `ERRORDETECTOR:SYMBOL:TEST:STATUS?` might return `:ERRORDETECTOR:SYMBOL:TEST:STATUS STOPPED;LOCKED;NO_SIGNAL;MAX_AP_OK`, indicating that all of the status for the symbol error test are STOPPED, LOCKED, NO_SIGNAL, and MAX_AP_OK.

ERRORDetector:SYMBOL:TEST:STATUS:LOCK? (Query Only)

This command queries the LOCK status for the symbol error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:STATUS:LOCK?`

Returns LOCKED or NOT_LOCKED for LOCK status for character error test.

Examples ERRORDETECTOR:SYMBOL:TEST:STATUS:LOCK? might return :ERRORDETECTOR:SYMBOL:TEST:STATUS:LOCK LOCKED, indicating that the recovered reference clock is LOCKED for the symbol error test.

ERRORDetector:SYMBOL:TEST:STATUS:MAX_AP? (Query Only)

This command queries the MAX_AP status for the symbol error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:SYMBOL:TEST:STATUS:MAX_AP?

Returns MAX_AP_OK or MAX_AP_FAIL for MAX_AP status for character error test.

Examples ERRORDETECTOR:SYMBOL:TEST:STATUS:MAX_AP? might return :ERRORDETECTOR:SYMBOL:TEST:STATUS:MAX_AP MAX_AP_OK, indicating that the MAX_AP status is ok for the symbol error test.

ERRORDetector:SYMBOL:TEST:STATUS:SIGNAL? (Query Only)

This command queries the SIGNAL status for the symbol error test.

NOTE. *Not available on some models.*

Group Error Detector

Syntax ERRORDetector:SYMBOL:TEST:STATUS:SIGNAL?

Returns SIGNAL or NO_SIGNAL for SIGNAL status for character error test.

Examples ERRORDETECTOR:SYMBOL:TEST:STATUS:SIGNAL? might return :ERRORDETECTOR:SYMBOL:TEST:STATUS:SIGNAL SIGNAL, indicating that the signal is connected for the symbol error test.

ERRORDetector:SYMBOL:TEST:STATUS:START? (Query Only)

This command queries the START status for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:STATUS:START?`

Returns `COUNTING` or `NOT_COUNTING` for `START` status for character error test.

Examples `ERRORDETECTOR:SYMBOL:TEST:STATUS:START?` might return `:ERRORDETECTOR:SYMBOL:TEST:STATUS:START COUNTING`, indicating that the symbol error test is comparing symbols and counting any errors that occur.

ERRORDetector:SYMBOL:TEST:TIME? (Query Only)

This command queries the elapsed time (in days, hours, minutes, and seconds) for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:TIME?`

Examples `ERRORDETECTOR:SYMBOL:TEST:TIME?` might return `:ERRORDETECTOR:SYMBOL:TEST:TIME 0,0,0,14`, indicating that the elapsed time is 0 days, 0 hours, 0 minutes, and 14 seconds for symbol error testing.

ERRORDetector:SYMBOL:TEST:TIME:DAYS? (Query Only)

This command queries the elapsed time days component for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:TIME:DAYS?`

Examples `ERRORDETECTOR:SYMBOL:TEST:TIME:DAYS?` might return `:ERRORDETECTOR:SYMBOL:TEST:TIME:DAYS 0`, indicating that the elapsed time days component is 0 for symbol error testing.

ERRORDetector:SYMBOL:TEST:TIME:HOURS? (Query Only)

This command queries the elapsed time hours component for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:TIME:HOURS?`

Examples `ERRORDETECTOR:SYMBOL:TEST:TIME:HOURS?` might return `:ERRORDETECTOR:SYMBOL:TEST:TIME:HOURS 0`, indicating that the elapsed time hours component is 0 for symbol error testing.

ERRORDetector:SYMBOL:TEST:TIME:MINUTES? (Query Only)

This command queries the elapsed time minutes component for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:TIME:MINUTES?`

Examples `ERRORDETECTOR:SYMBOL:TEST:TIME:MINUTES?` might return `:ERRORDETECTOR:SYMBOL:TEST:TIME:MINUTES 0`, indicating that the elapsed time minutes component is 0 for symbol error testing.

ERRORDetector:SYMBOL:TEST:TIME:SECOnds? (Query Only)

This command queries the elapsed time seconds component for symbol error testing.

NOTE. *Not available on some models.*

Group Error Detector

Syntax `ERRORDetector:SYMBOL:TEST:TIME:SECOnds?`

Examples `ERRORDETECTOR:SYMBOL:TEST:TIME:SECONDS?` might return `:ERRORDETECTOR:SYMBOL:TEST:TIME:SECONDS 25`, indicating that the elapsed time seconds component is 25 for symbol error testing.

ERRORDetector:TIMEformat

This command sets or queries error detector Elapsed Time Format as DDHHMMSS or Seconds.

NOTE. *Not available on some models.*

Group Error Detector

Syntax
```
ERRORDetector:TIMEformat {DDHHMMSS | SECONDS}
ERRORDetector:TIMEformat?
```

Arguments DDHHMMSS sets the time format to days, hours, minutes, and seconds.

SECONDS sets the error detector time format to seconds

Examples `ERRORDETECTOR:TIMEFORMAT SECONDS` sets the error detector time format to seconds.

`ERRORDETECTOR:TIMEFORMAT?` might return :ERRORDETECTOR:TIMEFORMAT DDHHMMSS indicating the error detector time format is set to days, hours, minutes, and seconds.

ERRORDetector:TYPe

This command sets or queries the error detector type.

NOTE. *Not available on some models.*

Group Error Detector

Syntax

```
ERRORDetector:TYPe {BIT|FRAME|SYMBOL|CHARACTER|PRBS7|PRBS9}
ERRORDetector:TYPe?
```

Arguments

`BIT` sets the error detector type to bit.

`FRAME` sets the error detector type to frame.

`SYMBOL` sets the error detector type to symbol.

`CHARACTER` sets the error detector type to character.

`PRBS7` sets the error detector type to PRBS7.

`PRBS9` sets the error detector type to PRBS9.

Examples `ERRORDETECTOR:TYPE BIT` sets the error detector type to bit.

`ERRORDETECTOR:TYPE?` might return `:ERRORDETECTOR:TYPE BIT`, indicating that the error detector type is bit.

*ESE

This command 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 more detailed discussion of the use of these registers, see Registers.

Group Status and Error

Syntax

```
*ESE <NR1>
*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 DESE the binary value 10111010.

*ESR? (Query Only)

This query-only command returns the contents of the Standard Event Status Register (SESR). *ESR? also clears the SESR (since reading the SESR clears it). For a more detailed discussion of the use of these registers, see Registers.

Group

Status and Error

Syntax

*ESR?

Related Commands

ALLEv?, *CLS, DESE, *ESE, EVENT?, EVMsg?, *SRE, *STB?

Examples

*ESR? might return *ESR 213, showing that the SESR contains the binary value 11010101.

EVENT? (Query Only)

This query-only command 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.

EVMsg? (Query Only)

This query-only command 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 with an explanatory message. For more 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 instrument. 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)

This query-only command returns the number of event codes that are in the Event Queue. This is useful when using the ALLEv? query, since it lets you know exactly how many events will be returned.

Group Status and Error

Syntax `EVQty?`

Related Commands ALLEv?, EVENT?, EVMsg?

Examples `EVQTY?` might return `:EVQTY 3`, indicating the number of event codes in the Event Queue.

EXPort

This command sends a copy of the waveform to the file path specified by EXPort:FILEName. The EXPort query returns image format and file information.

Group Hard Copy

Syntax
```
EXPort STARt
EXPort?
```

Arguments `STARt` initiates the export.

Examples `EXPORT START` initiates export to the file specified file by `EXPort:FILEName`.

`EXPORT?` returns the export formatting and file path. The query might return `:EXPORT:FILENAME "untitled"; FORMAT BMP;IMAGE NORMAL; PALETTE COLOR;VIEW FULLSCREEN "C\TekScope\Screen Captures\hcFILE.BMP"`.

EXPort:FILEName

This command sets or queries the file/path that will be sent export data on the next EXPort command.

Group Hard Copy

Syntax
```
EXPort:FILEName <file path>
EXPort:FILEName?
```

Related Commands HARDCopy

Arguments <file path> specifies that the hard copy is sent to the named file. <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>:<dir>/<filename>.<drive> and one or more <dir>s are optional. The file path cannot exceed 128 characters. If you do not specify the path with <drive>: and one or more <dir>s, the default location is "C:\Users\Public\Tektronix\TekScope\Screen Captures". While filename extensions are not required, they are highly recommended.

Examples EXPORT:FILENAME "TEK.BMP" sets TEK.BMP as the selected file name. The next export will be sent to this file in the default hard copy directory.

EXPORT:FILENAME? might return :EXPORT:FILENAME "C:\Users\Public\Tektronix\TekScope\Screen Captures\TEK.BMP" as the selected file.

EXPort:FORMat

This command sets or queries the image format for exporting waveforms to a file.

Group Hard Copy

Syntax
```
EXPort:FORMat {BMP|JPEG|PCX|PNG|TIFF}
EXPort:FORMat?
```

Arguments BMP specifies BMP image format.

JPEG specifies JPEG image format.

PCX specifies PCX image format.

PNG specifies PNG image format.

TIFF specifies TIFF image format.

Examples EXPORT:FORMAT BMP sets the export format to BMP.

EXPORT:FORMAT? might return :EXPORT:FORMAT JPEG, indicating that waveforms will be exported in the JPEG format.

EXPort:PALEtte

This command sets or queries the export color palette.

Group Hard Copy

Syntax

```
EXPort:PALEtte {COLOr|INKSaver|BLACKANDWhite}
EXPort:PALEtte?
```

Arguments

`COLOr` Hard copy output is color.

`INKSaver` Hard copy output saves ink.

`BLACKANDWhite` Hard copy output is black and white.

Examples

`EXPORT:PALETTE color` sets the export color palette to Color.

`EXPORT:PALETTE?` might return `:EXPORT PALETTE COLOR`, indicating the hard copy output will be in color.

EXPort:READOuts

This command sets or queries the area on the screen where the readout appear for export.

Group Hard Copy

Syntax

```
EXPort:READOuts {OFFGRAticule|ONGRAticule}
EXPort:READOuts?
```

Arguments

`OFFGRATICULE` places the readouts off the graticule area.

`ONGRATICULE` places the readouts on the graticule area for export.

Examples

`EXPORT:READOUTS OFFGRATICULE` places the readouts off the graticule area.

`EXPORT:READOUTS?` might return `:EXPORT READOUTS ONGRATICULE`, indicating the readout on the graticule area for export.

EXPort:VIEW

This command sets or queries the area of the screen to be exported.

Group Hard Copy

Syntax

```
EXPort:VIEW {FULLSCREEN|GRAticule|FULLNOmenu}
EXPort:VIEW?
```

Arguments

`FULLSCREEN` displays both the graticule and menu areas of the screen.

`GRAticule` displays only the graticule area of the screen.

`FULLNOmenu` displays the full screen but hides any menus or toolbars.

Examples

`EXPORT:VIEW FULLSCREEN` sets the exported view area to Full Screen.

`EXPORT:VIEW?` might return `:EXPORT:VIEW GRATICULE`, indicating that only the graticule area of the screen will be exported.

FACtory (No Query Form)

This command (no query form) resets the instrument to its factory default settings.

This command is equivalent to pressing the DEFAULT SETUP button located on the instrument front panel or selecting Recall Default Setup from the File menu.

This command resets the instrument settings to factory defaults:

- Clears the Event Status Enable Register.
- Clears the Service Request Enable Register.
- Sets the Device Event Status Enable Register to 255.
- Sets the Power On Status Clear Flag to TRUE.
- Purges all defined aliases.
- 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:

- The state of the GPIB (IEEE Std 488.1-1987) interface.
- The selected GPIB address.
- Calibration data that affects device specifications.
- Protected user data.
- Stored settings.
- The current password (if you set one).

Group Save and Recall

Syntax `FACtory`

Related Commands *PSC, *RCL, RECAll:SETUp, *RST, *SAV, SAVe:SETUp

Arguments None

Examples FACTORY resets the instrument to its factory default settings.

FASTAcq? (Query Only)

This query-only command returns the state of Fast Acquisitions. This command is equivalent to pressing the FASTACQ button on the front panel.

Group Acquisition

Syntax FASTAcq?

Examples FASTACQ? might return the following string for the current acquisition parameters: :FASTACQ:STATE 0.

FASTAcq:HIACQRATE

This command sets or queries the state of FastAcq optimization for capturing signal details with a higher sample rate.

Group Acquisition

Syntax
```
FASTAcq:HIACQRATE {<NR1>|OFF|ON}
FASTAcq:HIACQRATE?
```

Arguments <NR1> = 0 sets FastAcq optimization for capturing signal details with a higher sample rate; any other value sets FastAcq optimization for capturing rare events with more acquisitions.

OFF sets FastAcq optimization for capturing signal details with a higher sample rate.

ON sets FastAcq optimization for capturing rare events with more acquisitions.

Examples FASTACQ:HIACQRATE ON sets optimizes FastAcq for capturing rare events with more acquisitions.

FASTACQ:HIACQRATE? might return :FASTACQ:HIACQRATE 0, indicating that the FastAcq is optimized for capturing signal details with a higher sample rate.

FASTAcq:STATE

This command sets or queries the state of Fast Acquisitions. This command is equivalent to the FASTACQ button on the front panel.

Group Acquisition

Syntax
```
FASTAcq:STATE {<NR1>|ON|OFF}
FASTAcq:STATE?
```

Arguments <NR1> = 0 disables Fast Acquisitions mode; any other value enables Fast Acquisitions mode.

ON enables Fast Acquisitions mode.

OFF disables Fast Acquisitions mode.

Examples FASTACQ:STATE ON enables the Fast Acquisitions mode.

FASTACQ:STATE? might return :FASTACQ:STATE 1, indicating that Fast Acquisitions mode is currently active.

FILESystem? (Query Only)

This query-only command returns the directory listing of the current working directory. This query is the same as the FILESystem:DIR? query.

Group File System

Syntax
```
FILESystem?
```

Related Commands FILESystem:COPy, FILESystem:CWD, FILESystem:DELEte, FILESystem:DIR?, FILESystem:PRInt, FILESystem:READFile, FILESystem:REName, FILESystem:WRITEFile

Arguments None.

Examples `FILESYSTEM?` might return `:FILESYSTEM:DIR, "."," ..","myFile.txt","myWaveform.wfm".`

FILESystem:COPy (No Query Form)

This command (no query form) copies a named file to a new file. The new file might be in a totally separate directory than the old file. You can only copy one file at a time using this command. Wild card characters are not allowed.

Group File System

Syntax `FILESystem:COPy {<source file path>,<destination file path>}`

Related Commands FILESystem:CWD, FILESystem:DELEte

Arguments `<source 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.

`<destination 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:COPY "C:\MYDIR\TEK00001.SET", "C:\ANOTHERDIR\COPYOFTEK001.SET"` copies the file named TEK00001.SET, located in the MYDIR directory on the C drive to a file named COPYOFTEK001.SET in the ANOTHERDIR directory on the C drive.

FILESystem:CWD

This command sets or queries the current working directory for FILESystem GPIB commands. The default working directory is "C:\Users\Public\Tektronix\TekScope". Anytime that 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 instrument resets current working directory to the default directory (C:\Users\Public\Tektronix\TekScope) the next time the instrument is powered on or the next time you execute a file system command. The current working directory is retained between power cycles.

This command supports the permutations of file and directory names that are supported by Microsoft Windows:

- Relative path names; for example, ".\Temp"
- Absolute path names; for example, "C:\Windows\System"
- Implied relative path names; for example "newfile.text" becomes "C:\Users\Public\Tektronix\TekScope\newfile.txt" if the current working directory is "C:\Users\Public\Tektronix\TekScope"

Group File System

Syntax `FILESystem:CWD {<new working directory path>}`

Arguments `<new working directory path>` is a quoted string that defines the current working; a directory name can be up to 128 characters.

Examples `FILESYSTEM:CWD "C:\Users\Public\Tektronix\TekScope\Screen Captures"` sets the current working directory to Screen Captures.

`FILESYSTEM:CWD?` might return `:FILESYSTEM:CWD "C:\Users\Public\Tektronix\TekScope\Waveforms"`, indicating that the current working directory is set to Waveforms.

FILESystem:DELEte (No Query Form)

This command (no query form) deletes a named file. It does not delete directories. Use the FILESystem:RMDir command to delete a directory.

Group File System

Syntax `FILESystem:DELEte <file path>`

Related Commands FILESystem:COPy, FILESystem:CWD, FILESystem:RMDir

Arguments `<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:DELETE "NOT_MINE.SET"` deletes the file named NOT_MINE.SET from the current working directory.

FILESystem:DIR? (Query Only)

This query-only command returns a comma separated list of quoted strings. Each string contains the name of a file or directory in the current working directory.

Group File System

Syntax `FILESystem:DIR?`

Related Commands FILESystem:CWD, FILESystem:MKDir

Arguments None

Examples `FILESYSTEM:DIR?` returns a list of files and directories in the current working directory.

FILESystem:MKDir (No Query Form)

This command (no query form) creates a new directory.

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 "C:\NewDirectory"` creates the directory named *NewDirectory* at the root of the C drive.

These two commands create the directory MyNewSubDirectory within the existing directory MyDirectory at the root of the C drive:

```
FILESYSTEM:CWD "C:\MyDirectory"; FILESYSTEM:MKDIR
"MyNewSubDirectory"
```

This, of course, assumes that *C:\MyDirectory* already existed and was not a read-only directory.

FILESystem:PRInt (No Query Form)

This command (no query form) prints a named file to a named port. This command is the same as the FILESystem:READFile command.

This command is not IEEE 488.2 compliant for the following reasons:

- The command produces output on the 488.2 bus. Queries, not commands, are the only message units that should produce output on the 488.2 bus. Errors might not be recognized if they occur while data is being output in response to this command. This will be evident if a command is sent to the instrument before all of the data generated by this command is read. In such cases, you should send a device clear message to the instrument.
- The command does not use a standard 488.2 output format. The closest defined format is the indefinite format of ARBITRARY BLOCK PROGRAM DATA encoding. However, this command output omits the initial '#0' characters needed for that format.
- Newlines, ASCII value 10, can be included in the output data. Output on the bus can be terminated by newlines. The output of this command is only terminated by <EOI>.

Group File System

Syntax `FILESystem:PRInt <filepath>, GPIb`

Related Commands FILESystem:CWD, FILESystem:READFile

Arguments `<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.

`GPIb` specifies that the hard copy is sent out of the GPIB port. GPIb is the only port allowed for this command.

Examples `FILESYSTEM:PRINT "C:\TEK00000.IBM",GPIB` sends the file named TEK00000.IBM out of the GPIB port.

FILESystem:READFile (No Query Form)

This command (no query form) prints a named file to a named port. It is identical to the FILESystem:PRInt command.

This command is not IEEE 488.2 compliant for the following reasons:

- The command produces output on the 488.2 bus. Queries, not commands, are the only message units that should produce output on the 488.2 bus. Errors might not be recognized if they occur while data is being output in response to this command. This will be evident if a command is sent to the instrument before all of the data generated by this command is read. In such cases, you should send a device clear message to the instrument.
- The command does not use a standard 488.2 output format. The closest defined format is the indefinite format of ARBITRARY BLOCK PROGRAM DATA encoding. However, this command output omits the initial '#0' characters needed for that format.
- Newlines, ASCII value 10, can be included in the output data. Output on the bus can be terminated by newlines. The output of this command is only terminated by <EOI>.

Group File System

Syntax `FILESystem:READFile <filepath>`

Related Commands FILESystem:CWD, FILESystem:PRInt

Arguments `<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:READFILE "C:\TEK00000.IBM"` sends the file named TEK00000.IBM out of the GPIB port.

FILESystem:REName (No Query Form)

This command (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 "C:\TEK00000.SET","C:\MYSETTING.SET" gives the file named TEK00000.SET the new name of MYSETTING.SET. The file remains in the root directory on the C drive.

FILESystem:RMDir (No Query Form)

This command (no query form) deletes a named directory. The directory cannot contain any files. If there are files in the directory, they must first be deleted. 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 file name and path. If the file path is within the current working directory, you need only specify the file name.

Examples FILESYSTEM:RMDIR "C:\OldDirectory" removes the directory named OldDirectory from the root of the C drive.

FILESystem:WRITEFile (No Query Form)

This command (no query form) copies the block data from the GPIB port to a named file.

Group File System

Syntax FILESystem:WRITEFile <file path>,<data>

Related Commands FILESystem:CWD

Arguments <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.

<data> can be either DEFINITE LENGTH or INDEFINITE LENGTH ARBITRARY BLOCK PROGRAM DATA encoding as described in IEEE 488.2.

Examples FILESYSTEM:WRITEFILE "C:\SomeNewFile.txt",#221 1.23 2.54 8.22 -1.22<EOI> creates a file named "SomeNewFile.txt" on the root of the C drive, containing the string of 21 characters " 1.23 2.54 8.22 -1.22". This is an example of using the definite length arbitrary block.

FILESYSTEM:WRITEFILE "C:\SomeNewFile.txt",#0 This is a test<NL+EOI> creates a file named "SomeNewFile.txt" on the root of the C drive, containing the string "This is a test." This is an example of using the indefinite length arbitrary block. The newline character (0x0a) is required.

FILESYSTEM:WRITEFILE "C:\SomeNewFile.txt",#221 1.23 2.54 8.22 -1.22<NL+EOI> creates a file named "SomeNewFile.txt" on the root of the C drive, containing the string of 21 characters " 1.23 2.54 8.22 -1.22". Because the data argument is indefinite format arbitrary block program data, it can contain any 8-bit values, including newline characters (hexadecimal 0a). The fact that newline characters are allowed makes this command deviate from the strictest interpretation of the IEEE488.2 specification.

FPANel:PRESS (No Query Form)

This command turns off the displayed menu. It turns off everything except warning messages.

NOTE. *This command is only available on MSO/DPO5000/B Series instruments.*

Group Miscellaneous

Syntax FPANel:PRESS <menuoff>

Arguments menuoff turns off the menu.

Examples FPANEL:PRESS menuoff turns of the displayed menu.

HARDCopy

This command sends a copy of the screen display to the port specified by HARDCopy:PORT. This command is equivalent to pressing the PRINT button on the front panel. When printing to a file, the file format can be BMP, JPG, PNG, PCX or TIFF. The format of the saved screen capture is set by the EXPORT:FORMAT command. The file format setting is persistent, and will not be affected by a default setup or *RST command sent to the instrument.

The HARDCopy query returns the port and file path.

Group Hard Copy

Syntax

```
HARDCopy STARt
HARDCopy?
```

Related Commands *WAI, *CLS, EXPort:FORMat

Arguments STARt initiates a screen copy to a file or the default system printer, as specified by the :HARDCopy:PORT selection. The default system printer is set within the Windows operating system. If you need information about how to set the default system printer, refer to Microsoft Windows online help.

NOTE. *Use the *WAI command between HARDCopy STARt commands to ensure that the first hard copy is complete before starting another.*

Examples HARDCOPY START initiates a screen copy to a file or the default system printer, as specified by the :HARDCopy:PORT selection.

HARDCOPY? returns the hard copy file path and the selected port. The query might return :HARDCOPY:FILENAME "C:\Users\Public\Tektronix\TekScope\Screen Captures\hcFILE.BMP";PORT FILE.

HARDCopy:FILEName

This command sets or queries the file that will be sent hardcopy data on the next HARDCopy command (if the HARDCopy:PORT is set to FILE).

Group Hard Copy

Syntax

```
HARDCopy:FILEName <file path>
HARDCopy:FILEName?
```

Related Commands

HARDCopy, EXPort:FORMat

Arguments

<file path> specifies that the hard copy is sent to the named file. <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>:<dir>/<filename>.<drive> and one or more <dir>s are optional. The file path cannot exceed 128 characters. If you do not specify the path with <drive>: and one or more <dir>s, the default location is "C:\Users\Public\Tektronix\TekScope\Screen Captures". The filename extension is set by the EXPort:FORMat command.

Examples

HARDCOPY:FILENAME "TEK" sets TEK as the selected file name. The next hard copy will be sent to this file in the default hard copy directory, if the :HARDCopy:PORT selection is FILE.

HARDCOPY:FILENAME? might return :HARDCOPY:FILENAME "C:\Users\Public\Tektronix\TekScope\Screen Captures\TEK.BMP" as the selected file.

HARDCopy:LAYout

This command sets or queries the page orientation for hard copy.

Group

Hard Copy

Syntax

```
HARDCopy:LAYout {PORTRait|LANdscape}
HARDCopy:LAYout?
```

Arguments

PORTRait orients the screen image vertically on the printed page.

LANDscape orients the screen image horizontally on the printed page.

Examples

HARDCOPY:LAYOUT LANDSCAPE sets the hard copy page orientation to Landscape.

HARDCOPY:LAYOUT? might return :HARDCOPY:LAYOUT PORTRAIT, indicating that the hard copy page orientation is set to portrait.

HARDCopy:PALEtte

This command sets or queries the hard copy color palette.

Group Hard Copy

Syntax
```
HARDCopy:PALEtte {COLOr|BLACKANDWhite|INKSaver}
HARDCopy:PALEtte?
```

Arguments `COLOr` argument sets the hard copy output to color.

`BLACKANDWhite` argument sets the hard copy output to black and white.

`INKSaver` argument sets the hard copy output to save ink.

Examples `HARDCOPY:PALETTE COLOR` sets the hard copy output to Color.

`HARDCOPY:PALETTE?` might return `:HARDCOPY:PALETTE BLACKANDWHITE`, indicating that the output for hard copy will be in black and white.

HARDCopy:PORT

This command selects or returns whether the hard copy data will be sent to a file or printed on the next hard copy command (for example, the `HARDCopy STARt` command). This is equivalent to selecting Print in the File menu and then either choosing Print to file or specifying the default printer. If FILE is selected then the EXPORT setup commands are used to create the file. If PRINTER is selected, then the HARDCOPY commands are used to send the data to the printer.

Group Hard Copy

Syntax
```
HARDCopy:PORT {FILE|PRINTER}
HARDCopy:PORT?
```

Related Commands HARDCopy, HARDCopy:FILEName

Arguments `FILE` argument specifies that the hard copy is stored in the file specified in the HARDCopy:FILEName command.

`PRINTER` argument specifies that the hard copy is sent to the printer specified in the Print dialog box.

Examples `HARDCOPY:PORT FILE` specifies that the hard copy is stored in the file specified in the HARDCopy:FILEName command.

`HARDCOPY:PORT?` might return `:HARDCOPY:PORT FILE` indicating that hard copies are sent to the specified file.

The following commands sequence lets you to specify an inksaver data hard copy is stored in a file:

```
*RST
HARDCOPY:PORT FILE
HARDCOPY:PALETTE INKSAVER
HARDCOPY:FILENAME “FILE_INKSAVER”
HARDCOPY START
```

The following commands sequence lets you to specify a color data hard copy is stored in a file:

```
HARDCOPY:PALETTE COLOR
HARDCOPY:FILENAME :FILE COLOR”
HARDCOPY START
```

Both files will be the same since they are directed to a file. If they were directed to a printer then the printouts would be different.

HARDCopy:READOuts

This command sets or queries the area on a hard copy where the readout appears. This command is equivalent to selecting Page Setup from the File menu and setting the Readouts Below Graticule in the control window.

Group Hard Copy

Syntax

```
HARDCopy:READOuts {OFFGRAticule|ONGRAticule}
HARDCopy:READOuts?
```

Arguments

`OFFGRATICULE` places the readouts off the graticule area.

`ONGRATICULE` places the readouts on the graticule area on hardcopies.

HARDCopy:VIEW

This command sets or queries the area of the screen to be hard copied.

Group Hard Copy

Syntax

```
HARDCopy:VIEW {FULLSCREEN|GRAticule|FULLNOmenu}
HARDCopy:VIEW?
```

Arguments

FULLSCREEN sets the area to be hard copied to both the graticule and menu areas of the screen.

GRAticule sets the area to be hard copied to only the graticule area of the screen.

FULLNOmenu sets the area to be hard copied to full screen but hides any menus or toolbars.

Examples

HARDCOPY:VIEW FULLSCREEN sets the hard copy area to Full Screen.

HARDCOPY:VIEW? might return :HARDCOPY:VIEW GRATICULE, indicating that only the graticule area of the screen will be hard copied.

HDR

This command is identical to the HEADer query and is included for backward compatibility purposes.

Group

Miscellaneous

Syntax

```
HDR {<NR1>|OFF|ON}
HDR?
```

Related Commands

HEADer, VERBose

Arguments

<NR1> = 0 sets the Response Header Enable State to false; any other value sets this state to true, which causes the instrument to send headers on query responses.

OFF sets the Response Header Enable State to false. This causes the instrument 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 instrument to include headers on applicable query responses. You can then use the query response as a command.

Examples

HDR OFF specifies that the instrument omits headers on query responses, so that only the argument is returned.

HDR? might return :HEADER 1, indicating that the instrument is including headers on applicable query responses.

HEADer

This command sets or queries the Response Header Enable State that causes the instrument 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 make a corresponding change in the Response Header Enable State of the opposite interface (physical or virtual GPIB interface). Refer to Introduction for more information.*

Group Miscellaneous

Syntax

```
HEADer {<NR1>|OFF|ON}
HEADer?
```

Related Commands HDR, VERBose

Arguments

<NR1> = 0 sets the Response Header Enable State to false; any other value sets this state to true.

OFF sets the Response Header Enable State to false. This causes the instrument 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 instrument to include headers on applicable query responses. You can then use the query response as a command.

Examples

HEADER OFF specifies that the instrument omits headers on query responses, so that only the argument is returned.

HEADER? might return :HEADER 1, indicating that the instrument is including headers on applicable query responses.

HIStogram? (Query Only)

This query-only command returns all histogram parameters; it queries the state of all histogram parameters that the user can set. This command is equivalent to selecting Waveform Histograms from the Measure menu.

Group Histogram

Syntax `HIStogram?`

Related Commands MEASUrement:METHod

Examples `HISTOGRAM?` might return the histogram parameters as `:HISTOGRAM:BOXPCNT 30.0000,25.1000,70.0000,75.2000;DISPLAY LINEAR;STATE 1;FUNCTION VERTICAL;SIZE 2.0000;SOURCE CH1`.

HIStogram:BOX

This command defines or returns the left, top, right, and bottom boundaries of the histogram box, in source waveform coordinates. This command is equivalent to selecting Waveform Histograms from the Measure menu and then setting Limits for Left, Right, Top, and Bottom in the appropriate boxes. The command is similar to the HIStogram:BOXPcnt command except that command uses percentage coordinates to define the boundaries of the histogram box.

NOTE. *This command implicitly clears the histogram count and statistics for the histogram and histogram source data when HIStogram:STATE is turned on.*

Group Histogram

Syntax
```
HIStogram:BOX <NR3>,<NR3>,<NR3>,<NR3>
HIStogram:BOX?
```

Related Commands HIStogram:BOXPcnt

Arguments `<NR3>` specifies the left position of the histogram box in source waveform coordinates.

`<NR3>` specifies the top position of the histogram box in source waveform coordinates.

`<NR3>` specifies the right position of the histogram box in source waveform coordinates.

`<NR3>` specifies the bottom position of the histogram box in source waveform coordinates.

Examples `HISTOGRAM:BOX 1E-9, 0.250, 2E-9, 0.500` defines the coordinates of the histogram box in source waveform coordinates.

HISTOGRAM:BOX? might return :HISTOGRAM:BOX 1.0000E-09, 0.2500, 2.0000E-09, 0.5000, which identifies the coordinates of the histogram box in source waveform coordinates.

HIStogram:BOXPcnt

This command defines or queries the left, top, right, and bottom positions of the histogram box in percentage coordinates. The upper left has the value 0,0 and the lower right has the value 100, 100. Any value outside the range of 0 to 100 is clipped. This command is similar to the HIStogram:BOX command except that this command uses percentage coordinates to define the boundaries of the histogram box.

NOTE. *This command implicitly clears the histogram count and statistics for the histogram and histogram source data when HIStogram:STATE is turned on.*

Group Histogram

Syntax

```
HIStogram:BOXPcnt <NR2>, <NR2>, <NR2>, <NR2>
HIStogram:BOXPcnt?
```

Related Commands HIStogram:BOX

Arguments

<NR2> specifies the left position of the histogram box in percentage coordinates. The default value is 30.0.

<NR2> specifies the top position of the histogram box in percentage coordinates. The default value is 25.1.

<NR2> specifies the right position of the histogram box in percentage coordinates. The default value is 70.0.

<NR2> specifies the bottom position of the histogram box in percentage coordinates. The default value is 75.2.

Examples

HISTOGRAM:BOXPCNT 30.0,25.0,70.0,75.0 sets the left boundary of the histogram box to 30% of the graticule (3 divisions from the left edge); the top boundary to 25% of the graticule (0.25 × 8 = 2.0 divisions from the top edge); the right boundary to 70% of the graticule (7 divisions from the left edge); and the bottom boundary to 75% of the graticule (0.75 × 8 = 6.0 divisions from the top edge).

HISTOGRAM:BOXPCNT? might return :HISTOGRAM:BOXPCNT 30.0000,25.1000,70.0000,75.2000.

HIStogram:COUNt (No Query Form)

This command (no query form) clears the count and statistics for the histogram and the histogram source data. If the histogram is on, then counting restarts. This command is equivalent selecting Waveform Histograms from the Measure menu and then clicking the Reset button.

Group Histogram

Syntax `HIStogram:COUNt RESET`

Related Commands HIStogram:BOX, HIStogram:BOXPcnt

Examples `HISTOGRAM:COUNT RESET` clears the count and statistics for the histogram and the histogram source data.

HIStogram:DATa? (Query Only)

This query-only command returns an ASCII comma-separated list of histogram data numbers. This command returns 252 values for vertical histograms and 1000 values for horizontal histograms. The first and last value are not displayed, and are used for over range and under range tracking.

NOTE. *You must set the histogram state to ON before sending this query or the system will generate an error indicating: "Source waveform is not active".*

The Histogram:DATa? query corresponds to the Histogram Data (CSV) controls included in the instrument user interface. To find these controls, select either Edit > Copy > Setup > Measurements tab or the File > Save As > Histogram button.

This command returns only histogram data values separated by commas on a single line. The CSV file created by the user interface actions indicated above, writes out a line for each histogram data point, and contains the time relative to the trigger, a comma, and the histogram value.

Group Histogram

Syntax `HIStogram:DATa?`

Related Commands HIStogram:BOX, HIStogram:BOXPcnt, HIStogram:COUNt, HIStogram:DISplay, HIStogram:FUNCtion, HIStogram:MODe, HIStogram:SIZe, HIStogram:SOUrce, HIStogram:STATE

Examples `HISTOGRAM:DATA?` might return `:HISTOGRAM:DATA`

```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1245,933,903,1003,838,782,764,697,
687,620,582,566,530,595,557,552,518,524,606,
503,526,470,498,554,534,487,546,548,504,554,
513,566,639,508,643,704,662,769,785,806,922,
960,1153,1336,1478,1960,4515,17598,52184,
67040,50417,12885,4104,2382,1866,1699,1428,
1288,1271,1140,975,1024,894,841,841,678,841,
804,685,713,746,729,644,640,787,768,668,671,
661,745,755,692,882,772,802,862,770,854,799,
969,1102,1080,1040,1272,1329,1394,1760,1428,
2418,3313,4475,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0.
```

HIStogram:DISplay

This command sets the scaling of the histogram data display to be the count of each histogram bin or the log of that count. The default scaling is Linear.

The query form of this command returns either the histogram display scaling or the display status “off.”

This command is equivalent to selecting Waveform Histogram from the Measure menu, and then selecting Linear or Log in the Histogram Scaling group.

Group Histogram

Syntax

```
HIStogram:DISplay {OFF|LOG|LINEAr}
HIStogram:DISplay?
```

Related Commands HIStogram:MODe, HIStogram:STATE

Arguments `<OFF>` disables the histogram display but allows histogram data to accumulate. This argument provides for backward compatibility with older DPO Series instruments; it does not affect a change in the display scaling of the histogram. When queried, HIStogram:DISplay? will return `OFF` if the histogram state or mode is disabled using the HIStogram:MODe or HIStogram:STATE commands.

<LOG> sets the histogram display to logarithmic scaling.

<LINEAr> sets the histogram display to linear scaling. This is the default setting.

Examples

HISTOGRAM:DISPLAY LOG sets the histogram scaling to be the log count of each histogram bin.

HISTOGRAM:DISPLAY? might return :HISTOGRAM:DISPLAY LOG, indicating that the current histogram display scaling is the log count of each histogram bin.

HIStogram:FUNCtion

This command either selects the type of histogram to create or returns the current histogram type. This command is equivalent to selecting Waveform Histogram from the Measure menu and then choosing either Horizontal or Vertical from the Histogram Mode group box.

Group

Histogram

Syntax

```
HIStogram:FUNCtion {HORizontal|VERTical}
HIStogram:FUNCtion?
```

Related Commands

HIStogram:MODe, HIStogram:STATE

Arguments

HORizontal displays a horizontally positioned histogram that shows time distribution.

VERTical displays a vertically positioned histogram that shows a voltage distribution (or another distribution such as amperes).

Examples

HISTOGRAM:FUNCTION HORIZONTAL sets the histogram mode to the horizontal display mode.

HISTOGRAM:FUNCTION? might return :HISTOGRAM:FUNCTION VERTICAL, indicating that the vertical display mode is selected for the histogram.

HIStogram:MODe

This command selects the type of histogram to create or disables the histogram display. The query form either returns the current histogram type or that histogram display is disabled. This command is equivalent to selecting Waveform Histogram from the Measure menu and then choosing from the Histogram Mode group box.

HIStogram:MODe is functionally equivalent to the HIStogram:FUNCtion and HIStogram:STATE commands and is included for backward compatibility purposes.

Group Histogram

Syntax
```
HIStogram:MODe {HORizontal|VERTical|OFF}
HIStogram:MODe?
```

Related Commands HIStogram:FUNCtion, HIStogram:STATE, HIStogram:DISplay

Arguments `HORizontal` enables a horizontally positioned histogram that shows time distribution.

`VERTical` enables a vertically positioned histogram that shows a voltage distribution (or another distribution such as amperes).

`OFF` disables collection of the histogram measurement.

Examples `HISTOGRAM:MODE HORIZONTAL` sets the type of histogram created to horizontal, which displays a horizontally positioned histogram that shows time distribution.

`HISTOGRAM:MODE?` might return `:HISTOGRAM:MODE OFF`, indicating that histogram display is disabled.

HIStogram:SIZe

This command sets or queries the width or height of the histogram on the screen. This is equivalent to selecting Waveform Histograms from the Measure menu and then entering a value in the Histogram Size box.

Group Histogram

Syntax
```
HIStogram:SIZe <NR3>
HIStogram:SIZe?
```

Related Commands HIStogram:DISplay

Arguments `<NR3>` specifies the histogram size. The value can vary from 0.1 to 8.0 divisions in HORizontal mode and from 0.1 to 10.0 divisions in VERTical mode. Resolution is to the nearest pixel.

Examples

`HISTOGRAM:SIZE 2.0` sets the size of the histogram to 2.0 divisions.

`HISTOGRAM:SIZE?` might return `:HISTOGRAM:SIZE 2.0000E+01`, indicates the histogram size is set to 2.0 divisions.

HIStogram:SOUrce

This command sets or queries which source will be compared against the histogram box when the histogram testing is enabled. This is equivalent to selecting Waveform Histograms from the Measure menu and then choosing the desired waveform source. The waveform need not be displayed for histograms to run. You might want the channel displays disabled so you can see a full-screen histogram and not have waveform data confuse the display.

Group

Histogram

Syntax

```
HIStogram:SOUrce {CH<x>|MATH<x>|REF<x>}
HIStogram:SOUrce?
```

Related Commands

HIStogram:DISplay

Arguments

`CH<x>` selects a channel waveform as the source for the histogram. The x variable can be expressed as an integer ranging from 1 through 4.

`MATH<x>` selects a math waveform as the source for the histogram. The x variable can be expressed as an integer ranging from 1 through 4.

`REF<x>` selects a reference waveform as the source for the histogram. The x variable can be expressed as an integer ranging from 1 through 4.

Examples

`HISTOGRAM:SOURCE CH1` enables the Channel 1 waveform to be compared against the histogram box. The default time base, Main, is used for the histogram display.

`HISTOGRAM:SOURCE?` might return `:HISTOGRAM:SOURCE CH1`, indicating that the waveform for Channel 1 is the source for the histogram.

HIStogram:STATE

This command sets or queries whether the histogram calculations are enabled. This is equivalent to selecting Waveform Histograms from the Measure menu.

Group

Histogram

Syntax

```
HIStogram:STATE {<NR1>|ON|OFF}
HIStogram:STATE?
```

Related Commands HIStogram:MODe, MEASUrement:METHod

Arguments <NR1> = 0 disables histogram calculations; any other value enables histogram calculations.

<ON> enables histogram calculations.

<OFF> disables the histogram calculations.

Examples HISTOGRAM:STATE ON enables histogram calculations.

HISTOGRAM:STATE? might return :HISTOGRAM:STATE 0, indicating that histogram calculations are currently disabled.

HORizontal? (Query Only)

This query-only command returns all settings for the horizontal commands.

Group Horizontal

Syntax

```
HORizontal?
```

Examples HORIZONTAL? might return the following horizontal settings

```
:HORIZONTAL:DELAY:MODE 0;POSITION 50.0000;TIME
0.00000000000;:HORIZONTAL:MAIN:INTERPRATIO
1.0000;POSITION 50.0000;UNITS:STRING "s";:HORIZONTAL:ROLL
AUTO;MODE AUTO;MODE:AUTO:LIMITRECORDLEN
200000;:HORIZONTAL:MODE:RECORDLENGTH 1000;SAMPLERATE
40.0000E+9;SCAL.
```

HORizontal:ACQDURATION? (Query Only)

This query returns the timebase duration.

Group Horizontal

Syntax

```
HORizontal:ACQDURATION?
```

Returns <NR3> returns the duration of the acquisition.

Examples HORIZONTAL:ACQDURATION? might return :HORIZONTAL:ACQDURATION 5.0E-9, indicating the acquisition duration is 5.0 us.

HORizontal:ACQLENGTH? (Query Only)

This query returns the record length.

Group Horizontal

Syntax HORizontal:ACQLENGTH?

Returns <NR1> returns the record length.

Examples HORIZONTAL:ACQLENGTH? might return :HORIZONTAL:ACQLENGTH 50000, indicating the acquisition record length is 50000 points.

HORizontal:DIGital:MAGnivu:POSition? (Query Only)

NOTE. *Not available on some models.*

This command queries the horizontal position of the trigger in percent, within the MagniVu digital acquisition.

The MagniVu digital acquisition always includes the trigger. When the trigger position is adjusted near the edges of the display, the horizontal position of the MagniVu acquisition is adjusted to keep as much of the MagniVu acquisition in the display region as possible.

Group Horizontal

Syntax HORizontal:DIGital:MAGnivu:POSition?

Returns <NR3> specifies the horizontal position.

Examples `HORIZONTAL:DIGITAL:MAGNIVU:POSITION?` might return `:HORIZONTAL:DIGITAL:MAGNIVU:POSITION 50.0`, indicating that the horizontal position of the MagniVu digital acquisition is 50.0.

HORizontal:DIGital:RECOrdlength? (Query Only)

NOTE. *Not available on some models.*

NOTE. *The `HORizontal:DIGital:SAMPLERate?` returns the sample rate of the currently selected digital acquisition (either the Main values or the MagniVu values), depending on the ACQuire:MAGnivu setting.*

This command queries the record length of the digital acquisition.

Group Horizontal

Syntax `HORizontal:DIGital:RECOrdlength?`

Related Commands HORizontal:DIGital:SAMPLERate?

Examples `HORIZONTAL:DIGITAL:RECORDLENGTH?` might return `:HORIZONTAL:DIGITAL:RECORDLENGTH 125000`, indicating that the record length of the horizontal digital acquisition is set to 125000.

HORizontal:DIGital:RECOrdlength:MAGnivu? (Query Only)

NOTE. *Not available on some models.*

This command queries the record length of the MagniVu digital acquisition.

Group Horizontal

Syntax `HORizontal:DIGital:RECOrdlength:MAGnivu?`

Returns `<NR1>` returns the record length.

Examples `HORIZONTAL:DIGITAL:RECORDLENGTH:MAGNIVU?` might return `:HORIZONTAL:DIGITAL:RECORDLENGTH:MAGNIVU 10880`, indicating that the record length of the MagniVu digital acquisition is 10880.

HORizontal:DIGital:RECOrdlength:MAIn? (Query Only)

This command queries the record length of the main digital acquisition. This is the non-MagniVu, long record length acquisition.

Group Horizontal

Syntax `HORizontal:DIGital:RECOrdlength:MAIn?`

Returns `<NR1>` returns the record length.

Examples `HORIZONTAL:DIGITAL:RECORDLENGTH:MAIN?` might return `:HORIZONTAL:DIGITAL:RECORDLENGTH:MAIN 2500`, indicating that the record length of the main digital acquisition is 2500.

HORizontal:DIGital:SAMPLERate? (Query Only)

NOTE. *Not available on some models.*

This command queries the sample rate of the digital acquisition.

NOTE. *In the MSO/DPO5000/B Series instruments, the* `HORizontal:DIGital:SAMPLERate?` *returns the sample rate of the currently selected digital acquisition (either the Main values or the MagniVu values, depending on the ACQuire:MAGnivu setting.*

Group Horizontal

Syntax `HORizontal:DIGital:SAMPLERate?`

Related Commands HORizontal:DIGital:RECOrdlength?

Examples `HORIZONTAL:DIGITAL:SAMPLERATE?` might return `:HORIZONTAL:DIGITAL:SAMPLERATE 62.5000E+6`, indicating that the sample rate of the horizontal digital acquisition is set to 62.5000E+6.

HORizontal:DIGital:SAMPLERate:MAGnivu? (Query Only)

NOTE. *Not available on some models.*

This command queries the sample rate of the MagniVu digital acquisition.

Group Horizontal

Syntax `HORizontal:DIGital:SAMPLERate:MAGnivu?`

Returns `<NR3>` specifies the sample rate.

Examples `HORIZONTAL:DIGITAL:SAMPLERATE:MAGNIVU?` might return `:HORIZONTAL:DIGITAL:SAMPLERATE:MAGNIVU 16.5e9`, indicating that the sample rate of the MagniVu digital acquisition is 16.5e9.

HORizontal:DIGital:SAMPLERate:MAIn? (Query Only)

NOTE. *Not available on some models.*

This command queries the sample rate of the MagniVu digital acquisition.

Group Horizontal

Syntax `HORizontal:DIGital:SAMPLERate:MAIn?`

Returns `<NR3>` returns the sample rate.

Examples `HORIZONTAL:DIGITAL:SAMPLERATE:MAIN?` might return `:HORIZONTAL:DIGITAL:SAMPLERATE:MAIN 500e6`, indicating that the sample rate of the MagniVu digital acquisition is 500e6.

HORizontal:DIVisions? (Query Only)

This query-only command returns the number of graticule divisions over which the waveform is displayed.

Group Horizontal

Syntax `HORizontal:DIVisions?`

Examples `HORIZONTAL:DIVISIONS?` might return `:HORIZONTAL:DIVISIONS 10.0000`, indicating that the waveform is displayed across ten division.

HORizontal:FASTframe? (Query Only)

This query-only command returns all of the settings for the horizontal FastFrame commands.

Group Horizontal

Syntax `HORizontal:FASTframe?`

Examples `HORIZONTAL:FASTFRAME?` might return `:HORIZONTAL:FASTFRAME:STATE 1;READOUTS 0;SUMFRAME NONE;MULTIPLEFRAMES:MODE OVERLAY;FRAMESTART:CH1 1;CH2 1;CH3 1;CH4 1;MATH1 1;MATH2 1;MATH3 1;MATH4 1;REF1 1;REF2 1;REF3 1;REF4 1;:HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:NUMFRAMES:CH1 2;CH2 2;CH3 2;CH4 2;MATH1 2;MATH2 2;MATH3 2;MATH4 2;REF1 2;REF2 2;REF3 2;REF4 2;:HORIZONTAL:FASTFRAME:LENGTH 500;COUNT 2;SELECTED:CH1 2;CH2 2;CH3 2;CH4 2;MATH1 2;MATH2 2;MATH3 2;MATH4 2;REF1 2;REF2 2;REF3 2;REF4 2;:HORIZONTAL:FASTFRAME:REF:SOURCE CH4;FRAME 1;:HORIZONTAL:FASTFRAME:TRACK ALL`.

HORizontal:FASTframe:COUNt

This command sets or queries FastFrame frame count. This command is equivalent to selecting Fast Frame Setup from the Horiz/Acq menu and entering a value in the Frame Count box. FastFrame captures a series of triggered acquisitions with minimal intervening time.

Group Horizontal

Syntax
```
HORizontal:FASTframe:COUNt <NR1>
HORizontal:FASTframe:COUNt?
```

Related Commands HORizontal:MODE:RECOrdlength

Arguments <NR1> indicates the number of frames to acquire.

Examples HORIZONTAL:FASTFRAME:COUNT 2 sets up FastFrame mode to acquire two frames (segments) of data.

HORIZONTAL:FASTFRAME:COUNT? might return :HORIZONTAL:FASTFRAME:COUNT 4, indicating that FastFrame mode is currently set to acquire 4 frames of data.

HORizontal:FASTframe:MAXFRames? (Query Only)

This query only command returns the maximum number of FastFrame frames which can be acquired at the current frame length.

Group Horizontal

Syntax
```
HORizontal:FASTframe:MAXFRames?
```

Examples HORIZONTAL:FASTFRAME:MAXFRAMES? might return :HORIZONTAL:FASTFRAME:MAXFRAMES 3, indicating that the maximum number of FastFrame frames which can be acquired at the current frame length is 3.

HORizontal:FASTframe:MULtipleframes:FRAMESTart:<wfm>

This command sets or queries the start frame number on the specified waveform for the FastFrame multiple frames feature. Valid waveforms include CH<x>, MATH<x>, and REF<x>. The value of x can be an integer ranging from 1 through 4. The multiple frames feature supports displaying multiple frames in an overlaid manner.

Group Horizontal

Syntax

```
HORizontal:FASTframe:MULtipleframes:FRAMESTart:<wfm> <NR1>
HORizontal:FASTframe:MULtipleframes:FRAMESTart:<wfm>?
```

Related Commands

HORizontal:FASTframe:MULtipleframes:NUMFRames:<wfm>, HORizontal:FASTframe:MULtipleframes:MODe, HORizontal:FASTframe:COUNt

Arguments

<NR1> specifies the start frame number on the specified waveform.

Examples

HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:FRAMESTART:CH1 3 sets the start frame number to frame 3 on Channel 1.

HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:FRAMESTART:CH1? might return :HORIZONTAL:FASTFRAME:MULTIFRAMES:FRAMESTART:CH1 3, indicating that the start frame number is frame 3 for Channel 1.

HORizontal:FASTframe:MULtipleframes:MODe

This command sets or queries the mode for the FastFrame multiple frames feature. This feature displays multiple frames in an overlaid manner.

Group

Horizontal

Syntax

```
HORizontal:FASTframe:MULtipleframes:MODe
{OFF|OVERlay|ONLYOVERlay}
HORizontal:FASTframe:MULtipleframes:MODe?
```

Related Commands

HORizontal:FASTframe:MULtipleframes:FRAMESTart:<wfm>, HORizontal:FASTframe:MULtipleframes:NUMFRames:<wfm>, HORizontal:FASTframe:COUNt

Arguments

OFF turns off the multiple frames mode.

OVERlay sets the multiple frames mode to overlay.

Examples

HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:MODE OFF turns off the multiple frames mode.

HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:MODE? might return :HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:MODE OFF, indicating that the multiple frames mode is off.

HORizontal:FASTframe:MULtipleframes:NUMFRames:<wfm>

This command sets or queries the number of frames on the specified waveform for the FastFrame multiple frames feature. The multiple frames feature supports displaying multiple frames in an overlaid manner.

Valid waveforms include CH<x>, MATH<x> and REF<x>. The x variable can be expressed as an integer ranging from 1 through 4.

Group Horizontal

Syntax

```
HORizontal:FASTframe:MULtipleframes:NUMFRames:<wfm> <NR1>
HORizontal:FASTframe:MULtipleframes:NUMFRames:<wfm>?
```

Related Commands HORizontal:FASTframe:MULtipleframes:FRAMESTart:<wfm>, HORizontal:FASTframe:MULtipleframes:MODe, HORizontal:FASTframe:COUNt

Arguments <NR1> represents the number of frames on the specified waveform.

Examples HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:NUMFRAMES:CH1 3 sets the number of frames to 3 on Channel 1.

HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:NUMFRAMES:CH1? might return :HORIZONTAL:FASTFRAME:MULTIPLEFRAMES:NUMFRAMES:CH1 1, indicating that the number of frames on Channel 1 is 3.

HORizontal:FASTframe:REF:FRAme

This command sets or queries the FastFrame reference frame number. The Reference Frame number is used to calculate time differences for each frame and displays those differences in the graticule. This command is equivalent to selecting FastFrame Setup in the Horiz/Acq menu and entering a value in the Time Stamps/Frame box.

Group Horizontal

Syntax

```
HORizontal:FASTframe:REF:FRAme <NR1>
HORizontal:FASTframe:REF:FRAme?
```

Related Commands HORizontal:FASTframe:REF:SOUrce, HORizontal:FASTframe:TIMEStamp:DELTa:<wfm>?, HORizontal:FASTframe:TIMEStamp:REF?, HORizontal:FASTframe:TRACk

Arguments <NR1> specifies the FastFrame reference frame.

Examples HORIZONTAL:FASTFRAME:REF:FRAME 2 sets the FastFrame reference frame to 2.

HORIZONTAL:FASTFRAME:REF:FRAME? might return :HORIZONTAL:FASTFRAME:REF:FRAME 3, indicating that the FastFrame reference frame is currently set to 3.

HORizontal:FASTframe:REF:SOUrce

This command sets or queries FastFrame Reference waveform source. This is equivalent to selecting FastFrame Setup from the Horiz/Acq menu and choosing the reference source.

Group Horizontal

Syntax
```
HORizontal:FASTframe:REF:SOUrce <wfm>
HORizontal:FASTframe:REF:SOUrce?
```

Related Commands HORizontal:FASTframe:REF:FRAme, HORizontal:FASTframe:TIMEStamp:DELTa:<wfm>?, HORizontal:FASTframe:TIMEStamp:REF?, HORizontal:FASTframe:TRACk

Arguments <wfm> specifies the FastFrame Reference waveform source. Valid waveforms include CH<x> and MATH<x> The x variable can be expressed as an integer ranging from 1 through 4.

Examples HORIZONTAL:FASTFRAME:REF:SOURCE CH2 sets the horizontal FastFrame waveform reference source to Channel 2.

HORIZONTAL:FASTFRAME:REF:SOURCE? might return :HORIZONTAL:FASTFRAME:REF:SOURCE CH2, indicating that the FastFrame waveform source is currently set to Channel 2.

HORizontal:FASTframe:SELECTED:CH<x>

This command sets or queries the FastFrame selected frame number on the specified waveform.

Group Horizontal

Syntax

```
HORizontal:FASTframe:SELECTED:CH<x> <NRF>
HORizontal:FASTframe:SELECTED:CH<x>?
```

Arguments

<NRF> is the selected frame number.

Examples

HORIZONTAL:FASTFRAME:SELECTED:CH1 5 selects frame 5 on Channel 1.

HORIZONTAL:FASTFRAME:SELECTED:CH1? might return :HORIZONTAL:FASTFRAME:SELECTED:CH1 5, indicating frame 5 on Channel 1 is selected.

HORizontal:FASTframe:SELECTED:MATH<x>

This command sets or queries the FastFrame selected frame number on the specified waveform.

Group

Horizontal

Syntax

```
HORizontal:FASTframe:SELECTED:MATH<x> <NRF>
HORizontal:FASTframe:SELECTED:MATH<x>?
```

Arguments

<NRF> is the selected frame number.

Examples

HORIZONTAL:FASTFRAME:SELECTED:MATH2 4 selects frame 4 on Math 2 waveform.

HORIZONTAL:FASTFRAME:SELECTED:MATH1? might return :HORIZONTAL:FASTFRAME:SELECTED:MATH1 5, indicating that frame 5 on Math 1 waveform is selected.

HORizontal:FASTframe:SELECTED:REF<x>

This command sets or queries the FastFrame selected frame number on the specified waveform.

Group

Horizontal

Syntax

```
HORizontal:FASTframe:SELECTED:REF<x> <NRF>
HORizontal:FASTframe:SELECTED:REF<x>?
```

Arguments <NRF> is the selected frame number.

Examples HORIZONTAL:FASTFRAME:SELECTED:REF1 1 selects frame 1 on Reference 1 waveform.

HORIZONTAL:FASTFRAME:SELECTED:REF1? might return :HORIZONTAL:FASTFRAME:SELECTED:REF1 25, indicating frame 25 on Reference 1 waveform is the selected frame.

HORizontal:FASTframe:SELECTED:SOUrce

This command sets or queries the FastFrame source waveform. This is equivalent to selecting FastFrame Setup from the Horiz/Acq menu, and then choosing the waveform source. Valid waveforms include CH<x> and MATH<x>. The x variable can be expressed as an integer ranging from 1 through 4.

Group Horizontal

Syntax
```
HORizontal:FASTframe:SELECTED:SOUrce <NR1>
HORizontal:FASTframe:SELECTED:SOUrce?
```

Related Commands HORizontal:FASTframe:TRACk, HORizontal:FASTframe:TIMEStamp:DELTa:<wfm>?, HORizontal:FASTframe:TIMEStamp:REF?

Arguments <NR1> specifies the selected frame number on the specified waveform.

Examples HORIZONTAL:FASTFRAME:SELECTED:SOURCE CH2 sets the FastFrame source to Channel 2.

HORIZONTAL:FASTFRAME:SELECTED:SOURCE? might return :HORIZONTAL:FASTFRAME:SELECTED:SOURCE CH1, indicating that the selected source is Channel 1.

HORizontal:FASTframe:SEQuence

This command sets or queries the FastFrame single-sequence mode stop condition.

Group Horizontal

Syntax

```
HORizontal:FASTframe:SEQuence {FIRst|LAST}
HORizontal:FASTframe:SEQuence?
```

Arguments

`FIRst` sets single sequence to stop after n frames.

`LAST` sets single sequence to stop manually.

Examples

`HORIZONTAL:FASTFRAME:SEQUENCE First` sets single-sequence mode to stop after n frames.

`HORIZONTAL:FASTFRAME:SEQUENCE?` might return `:HORIZONTAL:FASTFRAME:SEQUENCE LAST`, indicating the FastFrame single-sequence mode is stop manually.

HORizontal:FASTframe:SINGLEFramemath

This command sets or queries FastFrame single frame math.

Group Horizontal

Syntax

```
HORizontal:FASTframe:SINGLEFramemath {<NR1>|OFF|ON}
HORizontal:FASTframe:SINGLEFramemath?
```

Arguments

`<NR1>` = 0 disables the function; any other value enables it.

`OFF` disables the function.

`ON` enables the function.

Examples

`HORizontal:FASTframe:SINGLEFramemath ON` turns single frame math on.

`HORizontal:FASTframe:SINGLEFramemath?` might return `:HORIZONTAL:FASTFRAME:SINGLEFRAMEMATH 0`, indicating that single frame math is off.

HORizontal:FASTframe:SIXteenbit

This command sets or queries FastFrame sixteen bit.

Group Horizontal

Syntax

```
HORizontal:FASTframe:SIXteenbit {<NR1>|OFF|ON}
HORizontal:FASTframe:SIXteenbit?
```

Arguments

<NR1> = 0 disables the function; any other value enables it.

OFF disables the function. When OFF, fastframe data is 8 bit.

ON enables the function. When ON and when the summary frame is average, the data for the averaged summary frame is 16 bit.

Examples

HORizontal:FASTframe:SIXteenbit 0 disables FastFrame sixteen bit.

HORizontal:FASTframe:SIXteenbit? might return :HORIZONTAL:FASTFRAME:SIXTEENBIT 1, indicating that FastFrame sixteen bit is on.

HORizontal:FASTframe:STATE

This command sets or queries the state of FastFrame acquisition. This is equivalent to setting FastFrame to On in the FastFrame Setup menu. FastFrame lets users capture a series of triggered acquisitions with minimal time between them.

The digitizing instrument in FastFrame mode is ready to accept a continuous burst of triggers 400 ms after the controller sends the ACQuire:STATE RUN command.

Group

Horizontal

Syntax

```
HORizontal:FASTframe:STATE {<NR1>|ON|OFF}
HORizontal:FASTframe:STATE?
```

Related Commands

ACQuire:STATE

Arguments

<NR1> = 0 turns off horizontal FastFrame; any other value turns on horizontal FastFrame.

<ON> turns on horizontal FastFrame.

<OFF> turns off horizontal FastFrame.

Examples

HORIZONTAL:FASTFRAME:STATE ON turns on horizontal FastFrame.

HORIZONTAL:FASTFRAME:STATE? might return :HORIZONTAL:FASTFRAME:STATE 0, indicating that the current state of FastFrame is off.

HORizontal:FASTframe:SUMFrame

This command sets or queries the summary frame mode. When ENVelope is selected, the last frame in a FastFrame acquisition is an envelope of all the prior frames in the acquisition. When AVErage is selected, the last frame is replaced with a frame that is the computed average of all the prior frames in the acquisition. For the summary frame control to be active, the number of frames must be two or greater.

Group Horizontal

Syntax

```
HORizontal:FASTframe:SUMFrame {NONe|AVErage|ENVelope}
HORizontal:FASTframe:SUMFrame?
```

Arguments

`NONE` turns off the summary mode for FastFrame. This is the default setting.

`AVErage` argument displays the last frame in a FastFrame acquisition as a frame that is the computed average of all the prior frames in the acquisition.

`ENVelope` argument displays the last frame in a FastFrame acquisition as an envelope of all the prior frames in the acquisition.

Examples

`HORIZONTAL:FASTFRAME:SUMFRAME NONE` turns off the summary frame feature.

`HORIZONTAL:FASTFRAME:SUMFRAME?` might return `:HORIZONTAL:FASTFRAME:SUMFRAME AVERAGE`, indicating that the summary mode for FastFRame is set to AVERAGE and will display the last frame in the acquisition as the average of all the prior frames in the acquisition.

HORizontal:FASTframe:TIMEStamp:ALL:<wfm>? (Query Only)

This query-only command returns the frame number and time stamp for each frame between requested frames, inclusive, within the specified waveform. Argument order is unimportant. Valid waveforms include CH<x>, MATH<x> and REF<x>. The x variable can be expressed as an integer ranging from 1 through 4.

The format of the output is as follows:

DD MON YYYY HH:MM:SS.xxx xxx xxx xxx

- DD MON YYYY is day, month, and year
- HH:MM:SS is hours, minutes, and seconds
- .xxx xxx xxx xxx is fractions of a second to picoseconds

Group Horizontal

Syntax `HORizontal:FASTframe:TIMEStamp:ALL:<wfm>? <NR1>, <NR1>`

Arguments <NR1> specifies the selected “from” frame number within the specified waveform.

<NR1> specifies the selected “to” frame number within the specified waveform.

Examples `HORIZONTAL:FASTFRAME:TIMESTAMP:ALL:CH2? 4,1` might return `:HORIZONTAL:FASTFRAME:TIMESTAMP:ALL:CH1 4,1,“02 Mar 2000 20:10:54.542 037 272 620”` This is the list of time stamps from frame 1 through frame 4 on Channel 1.

HORizontal:FASTframe:TIMEStamp:BETWeen:<wfm>? (Query Only)

This query-only command returns the relative trigger for the delta time between the specified frames, within the specified waveform. Valid waveforms include CH<x>, MATH<x> and REF<x>. The x variable can be expressed as an integer ranging from 1 through 4.

The format of the delta time output is as follows:

HH:MM:SS.xxx xxx xxx xxx

- HH:MM:SS is hours, minutes, and seconds
- .xxx xxx xxx xxx is fractions of a second to picoseconds

Group Horizontal

Syntax `HORizontal:FASTframe:TIMEStamp:BETWeen:<wfm>? <NR1>, <NR1>`

Arguments <NR1> specifies the selected “from” frame number within the specified waveform.

<NR1> specifies the selected “to” frame number within the specified waveform.

Examples `HORIZONTAL:FASTFRAME:TIMESTAMP:BETWEEN:CH1? 5,2` might return `:HORIZONTAL:FASTFRAME:TIMESTAMP:BETWEEN:CH1 5,2,"00:00:00.010 000 540 624"`. This is the delta of Channel 1's Time Stamp frame 5 - Time Stamp frame 2 (TS[5]-TS[2]).

HORizontal:FASTframe:TIMEStamp:DELTa:<wfm>? (Query Only)

This query-only command returns the relative time between the triggers of the FastFrame Selected and the FastFrame Reference, within the specified waveform. Valid waveforms include CH<x>, MATH<x>, and REF<x>. The x variable can be expressed as an integer ranging from 1 through 4.

The format of the output is as follows:

HH:MM:SS.xxx xxx xxx xxx

- HH:MM:SS is hours, minutes, and seconds
- .xxx xxx xxx xxx is fractions of a second to picoseconds

Group Horizontal

Syntax `HORizontal:FASTframe:TIMEStamp:DELTa:<wfm>?`

Examples `HORIZONTAL:FASTRAME:TIMESTAMP:DELTA:CH1?` might return `:HORIZONTAL:FASTFRAME:TIMESTAMP:DELTA:CH1 "00:00:00.006 000 306 556"` This is the delta time stamp of FastFrame Selected Frame on Channel 1 - FastFrame Reference Frame (TS[C1 Selected]-TS[Reference]).

HORizontal:FASTframe:TIMEStamp:FRAMe:<wfm>? (Query Only)

This query-only command returns the absolute trigger date and time for the specified frame and waveform. Valid waveforms include CH<x>, MATH<x> and REF<x>. The x variable can be expressed as an integer ranging from 1 through 4.

The format of the output is as follows:

DD MON YYYY HH:MM:SS.xxx xxx xxx xxx

- DD MON YYYY is day, month, and year
- HH:MM:SS is hours, minutes, and seconds
- .xxx xxx xxx xxx is fractions of a second to picoseconds

Group Horizontal

Syntax `HORizontal:FASTframe:TIMEStamp:FRAMe:<wfm>? <NR1>`

Arguments `<NR1>` is the frame number for which the timestamp will be returned on the specified waveform.

Examples `HORIZONTAL:FASTFRAME:TIMESTAMP:FRAME:CH1? 10` might return `:HORIZONTAL:FASTFRAME:TIMESTAMP:FRAME:CH1 10,"02 Mar 2000 20:10:54.536 036 928 432"` which is the time stamp of Channel 1, frame 10.

HORizontal:FASTframe:TIMEStamp:REF? (Query Only)

This query-only command returns the absolute trigger date and time for FastFrame reference.

The format of the output is as follows:

DD MON YYYY HH:MM:SS.xxx xxx xxx xxx

- DD MON YYYY is day, month, and year
- HH:MM:SS is hours, minutes, and seconds
- .xxx xxx xxx xxx is fractions of a second to picoseconds

Group Horizontal

Syntax `HORizontal:FASTframe:TIMEStamp:REF?`

Related Commands HORizontal:FASTframe:REF:FRAme, HORizontal:FASTframe:REF:SOUrce

Examples `HORIZONTAL:FASTFRAME:TIMESTAMP:REF?` might return `:HORIZONTAL:FASTFRAME:TIMESTAMP:REF "02 Mar 2000 20:10:54.533 036 838 784"`. This is the time stamp of FastFrame Reference.

HORizontal:FASTframe:TIMEStamp:SELECTED:<wfm>? (Query Only)

This query-only command returns the absolute trigger date and time for the FastFrame Selected, within the specified waveform. Valid waveforms include CH<x>, MATH<x> and REF<x>. The x variable can be expressed as an integer ranging from 1 through 4.

The format of the output is as follows:

DD MON YYYY HH:MM:SS.xxx xxx xxx xxx

- DD MON YYYY is day, month, and year
- HH:MM:SS is hours, minutes, and seconds
- .xxx xxx xxx xxx is fractions of a second to picoseconds

Group Horizontal

Syntax `HORizontal:FASTframe:TIMEStamp:SELECTED:<wfm>?`

Related Commands HORizontal:FASTframe:SELECTED:SOUrce

Examples `HORIZONTAL:FASTFRAME:TIMESTAMP:SELECTED:REF1?` might return `:HORIZONTAL:FASTFRAME:TIMESTAMP:SELECTED:REF1 "02 MAR 2000 20:10:54:539 037 145 340"`. This is the time stamp of FastFrame Selected on Reference 1 waveform.

HORizontal:FASTframe:TRACk

This command sets up or queries the state of FastFrame tracking feature. This command is equivalent to selecting FastFrame Setup from the Horiz/Acq menu and then clicking the desired Frame Tracking state.

When FastFrame Track is set to “live”, the channel and math waveforms are locked together. Adjusting a channel waveform also adjusts a related math waveform. All reference waveforms are also locked together but they are separate from channel and math waveforms.

For example, when you set the Selected Frame Source Ch1 to Frame 3, then Selected Frame Ch2, Ch3, Ch4, Math1, Math2, Math3 and Math4 are also set to Frame 3.

When you set the Selected Frame Source Ref1 to Frame 2, then Selected Frame Ref2, Ref3 and Ref4 are also set to Frame 2. If the Frame Tracking is set to Live, changing Selected Frame Ch1 will not affect the Selected Frame Ref1 frame of the Reference Frame setting.

When FastFrame Track is set to “all”, the channel, math and reference waveforms are locked together. Adjusting a channel waveform also adjusts the related math and reference waveforms.

For example, when you set the Selected Frame Source Ch1 to Frame 3, then Selected Frame Ch2, Ch3, Ch4, Math1, Math2, Math3, Math4, Ref1, Ref2, Ref3 and Ref4 are also set to Frame 3.

Group Horizontal

Syntax
```
HORizontal:FASTframe:TRACk {LIVE|ALL}
HORizontal:FASTframe:TRACk?
```

Related Commands HORizontal:FASTframe:REF:FRAme, HORizontal:FASTframe:REF:SOUrce, HORizontal:FASTframe:SELECTED:SOUrce

Arguments `LIVE` sets FastFrame Track to Live.

`ALL` sets FastFrame Track to All.

Examples `HORIZONTAL:FASTFRAME:TRACK LIVE` sets FastFrame Track to live.

`HORIZONTAL:FASTFRAME:TRACK?` might return `:HORIZONTAL:FASTFRAME:TRACK ALL`, indicating that all FastFrame Selected Frames and the FastFrame Reference will be locked together.

HORizontal:FASTframe:XZEro:ALL:CH<x>? (Query Only)

This command queries the time from the trigger to the trigger sample of the specified frames on the specified channel.

Group Horizontal

Syntax `HORizontal:FASTframe:XZEro:ALL:CH<x>? <NR1>,<NR1>`

Arguments `<NR1>` specifies the first and last frame of a range of frames.

Examples `HORIZONTAL:FASTFRAME:XZERO:ALL:CH1? 1,4` might return `:HORIZONTAL:FASTFRAME:XZERO:ALL:CH1 1,4,"1: 2.5265E-009","2: 3.9010E-010","3: 3.8285E-009","4: 1.3237E-009"`, giving the first 4 XZERO times for Channel 1.

HORizontal:FASTframe:XZEro:ALL:REF<x>? (Query Only)

This command queries the time from the trigger to the trigger sample of the specified frames on the specified reference.

Group Horizontal

Syntax `HORizontal:FASTframe:XZEro:ALL:REF<x>? <NR1>,<NR1>`

Arguments `<NR1>` specifies the first and last frame of a range of frames.

Examples HORIZONTAL:FASTFRAME:XZERO:ALL:REF1? 1,3 might return :HORIZONTAL:FASTFRAME:XZERO:ALL:REF1 1,3,"1: 2.5265E-009","2: 3.9010E-010","3: 3.8285E-009", giving the first 3XZERO times for Reference 1.

HORizontal:FASTframe:XZEro:FRAme:CH<x>? (Query Only)

This command queries the time from the trigger to the trigger sample of the specified frame on the specified channel.

Group Horizontal

Syntax HORizontal:FASTframe:XZEro:FRAme:CH<x>? <NR1>

Arguments <NR1> specifies a frame on the specified channel.

Examples HORIZONTAL:FASTFRAME:XZERO:FRAME:CH1? 3 might return :HORIZONTAL:FASTFRAME:XZERO:FRAME:CH1 3,3.8285E-9, indicating the time from the trigger to the start of frame 3 on Channel 1 is 3.83 ns.

HORizontal:FASTframe:XZEro:FRAme:REF<x>? (Query Only)

This command queries the time from the trigger to the trigger sample of the specified frame on the specified reference.

Group Horizontal

Syntax HORizontal:FASTframe:XZEro:FRAme:REF<x>? <NR1>

Arguments <NR1> specifies a frame on the specified reference.

Examples HORIZONTAL:FASTFRAME:XZERO:FRAME:REF1? 3 might return :HORIZONTAL:FASTFRAME:XZERO:FRAME:REF1 3,3.8285E-9, indicating the time from the trigger to the start of frame 3 on Reference 1 is 3.83 ns.

HORizontal:FASTframe:XZEro:REF? (Query Only)

This command sets or queries the time from the trigger to the trigger sample on the reference waveform.

Group Horizontal

Syntax `HORizontal:FASTframe:XZEro:REF?`

Examples `HORIZONTAL:FASTFRAME:XZERO:REF1?` might return `:HORIZONTAL:FASTFRAME:XZERO:REF 35.1000E-12`, indicating that the Reference 1 waveform trigger sample occurred 35.1 ps after the trigger.

HORizontal:FASTframe:XZEro:SELECTED:CH<x>? (Query Only)

This command sets or queries the time from the trigger to the trigger sample on the selected channel. CH<x> can be CH1, CH2, CH3, or CH4.

Group Horizontal

Syntax `HORizontal:FASTframe:XZEro:SELECTED:CH<x>?`

Examples `HORIZONTAL:FASTFRAME:XZERO:SELECTED:CH1?` might return `:HORIZONTAL:FASTFRAME:XZERO:SELECTED:CH1 35.1000E-12`, indicating that the selected Channel 1 trigger sample occurred 35.1 ps after the trigger.

HORizontal:FASTframe:XZEro:SELECTED:REF<x>? (Query Only)

This command sets or queries the time from the trigger to the trigger sample on the selected reference waveform. REF<x> can be REF1, REF2, REF3, or REF4.

Group Horizontal

Syntax `HORizontal:FASTframe:XZEro:SELECTED:REF<x>?`

Examples `HORIZONTAL:FASTFRAME:XZERO:SELECTED:REF1?` might return `:HORIZONTAL:FASTFRAME:XZERO:SELECTED:REF1 35.1000E-12`, indicating that the selected Reference 1 waveform trigger sample occurred 35.1 ps after the trigger.

HORizontal:MAIn? (Query Only)

This query-only command returns the time per division of the time base. This command is equivalent to selecting Position/Scale from the Horiz/Acq menu.

Group Horizontal

Syntax `HORizontal:MAIn?`

Related Commands HORizontal:MODE:SCAle

Examples `HORIZONTAL:MAIN?` might return `:HORIZONTAL:MAIN:INTERPRATIO 1.0000;SCALE 40.0000E-9;POSITION 50.0000;SAMPLERATE 1.2500E+9;UNITS:STRING "s"`.

HORizontal[:MAIn]:DELay:MODe

This command sets or queries the time base trigger delay mode. This command is equivalent to choosing Delay Mode On from the Horiz/Acq menu.

Group Horizontal

Syntax
```
HORizontal[:MAIn]:DELay:MODe {<NR1>|ON|OFF}
HORizontal[:MAIn]:DELay:MODe?
```

Related Commands HORizontal[:MAIn]:DELay:TIMe

Arguments `<NR1>` = 0 disables the time base trigger delay mode, any other value enables the time base trigger delay mode.

`ON` enables the time base trigger delay mode.

`OFF` disables the time base trigger delay mode.

Examples `HORIZONTAL:DELAY:MODE ON` enables the time base trigger delay mode.

`HORIZONTAL:DELAY:MODE?` might return `:HORIZONTAL:DELAY:MODE 1`, indicating that the time base trigger delay mode is currently enabled.

HORizontal[:MAIn]:DELay:POSition

This command sets or queries the time base position when Horizontal Delay Mode is turned on. This command is equivalent to selecting Horizontal/Acquisition Setup from the Horiz/Acq menu and then entering a Ref Point value.

Group Horizontal

Syntax

```
HORizontal[:MAIn]:DELay:POSition <NR3>
HORizontal[:MAIn]:DELay:POSition?
```

Related Commands HORizontal[:MAIn]:DELay:TIMe

Arguments <NR3> is from 0 to ≈100 and is the percentage of the waveform that is displayed left of the center graticule.

NOTE. *The upper limit of the waveform position is slightly limited by a value that is determined from the record length (upper limit = 100 - 1/record length).*

Examples HORIZONTAL:DELAY:POSITION 5E+1 sets the time base trigger delay position to 50 percent.

HORIZONTAL:DELAY:POSITION? might return :HORIZONTAL:DELAY:POSITION 5.0000E+01, indicating that the time base trigger delay position is currently set to 50 percent.

HORizontal[:MAIn]:DELay:TIMe

This command sets or queries the time base trigger delay time. This command is equivalent to selecting Position/Scale from the Horiz/Acq menu and choosing a value for Horiz Delay.

Group Horizontal

Syntax

```
HORizontal[:MAIn]:DELay:TIMe <NR3>
HORizontal[:MAIn]:DELay:TIMe?
```

Related Commands HORizontal[:MAIn]:DELay:MODe, HORizontal[:MAIn]:DELay:POSition

Arguments <NR3> specifies the time base trigger delay time setting, typically represented in seconds.

Examples HORIZONTAL:DELAY:TIME 5.0E-3 sets the time base trigger delay time to 5 ms.

HORIZONTAL:DELAY:TIME? might return :HORIZONTAL:DELAY:TIME 5.0000E-05, indicating that the time delay setting is 5 µs.

HORizontal:MAIn:INTERPRatio? (Query Only)

This query-only command returns the Horizontal Main time base interpolation ratio. The ratio is useful for calculating the maximum sampling rate when users request sampling rates that are faster than the maximum rate specified by the acquisition hardware.

The default maximum sampling rate as specified by the acquisition hardware is 1.

An interpolation ratio other than 1 means that displayed and retrieved waveforms contain additional points that are derived through interpolation. At such times, the retrieved waveform data is actually 16-bit data. You should set the data width accordingly.

Group Horizontal

Syntax
```
HORizontal:MAIn:INTERPRatio?
```

Examples HORIZONTAL:MAIN:INTERPRATIO? might return :HORIZONTAL:MAIN:INTERPRATIO 10.0.

HORizontal[:MAIn]:POSition

Group Horizontal

Syntax
```
HORizontal[:MAIn]:POSition <NR3>
HORizontal[:MAIn]:POSition?
```

Related Commands HORizontal[:MAIn]:DELay:MODe

Arguments <NR3> argument can range from 0 to ≈100 and is the percentage of the waveform that is displayed left of the center graticule.

NOTE. The upper limit of the waveform position is slightly limited by a value that is determined from the record length (upper limit = 100 - 1/record length).

Examples `HORIZONTAL:MAIN:POSITION 5.000E+01` sets the horizontal position of the waveform such that 10 percent of the waveform is to the left of center of the screen.

`HORIZONTAL:MAIN:POSITION?` might return `:HORIZONTAL:MAIN:POSITION 5.0000E+01`, indicating that the horizontal position of the waveform on the screen is currently set to 50 percent.

HORizontal:MAIn:UNIts

This command sets or queries the units for the horizontal time base. It is equivalent to setting the HORizontal:MAIn:UNIts:STRing.

Group Horizontal

Syntax
```
HORizontal:MAIn:UNIts <QString>
HORizontal:MAIn:UNIts?
```

Related Commands HORizontal:MAIn:UNIts:STRing

Arguments `<QString>` is the time base units string.

Examples `HORIZONTAL:MAIN:UNITS "LUM"` sets the time base multiplier to 5.5, the units to lumens and the time base units offset to 2.9.

`HORIZONTAL:MAIN:UNITS?` might return `:HORIZONTAL:MAIN:UNITS:STRING` “lum”.

HORizontal:MAIn:UNIts:STRing

This command sets or queries the units string for the horizontal time base.

Group Horizontal

Syntax
```
HORizontal:MAIn:UNIts:STRing <QString>
HORizontal:MAIn:UNIts:STRing?
```

Related Commands HORizontal:MAIn:UNIts

Arguments <QString> is the time base units string.

Examples HORIZONTAL:MAIN:UNITS:STRING "LUM" sets the units string for the time base trigger delay to lumens.

HORIZONTAL:MAIN:UNITS:STRING? might return :HORIZONTAL:MAIN:UNITS:STRING "lum", indicating that the units string for the time base trigger delay is set to lumens.

HORizontal:MODE

This command set or queries the horizontal mode. Auto mode is the factory default.

Group Horizontal

Syntax
```
HORizontal:MODE {AUTO|CONStant|MANual}
HORizontal:MODE?
```

Related Commands HORizontal:MODE:AUTO:LIMITrecordlen, HORizontal:MODE:RECOrdlength, HORizontal:MODE:SAMPLERate, HORizontal:MODE:SCAle

Arguments AUTO selects the automatic horizontal model. Auto mode attempts to keep record length constant as you change the time per division setting. Record length is read only.

CONSTANT selects the constant horizontal model. Constant mode attempts to keep sample rate constant as you change the time per division setting. Record length is read only.

MANUAL selects the manual horizontal model. Manual mode lets you change sample mode and record length. Time per division or Horizontal scale is read only.

Examples HORIZONTAL:MODE AUTO sets the horizontal mode to auto.

HORIZONTAL:MODE? might return :HORIZONTAL:MODE MANUAL, indicating that the horizontal mode is manual.

HORizontal:MODE:AUTO:LIMITrecordlen

This command sets or queries the record length limit used by the auto horizontal mode.

Group Horizontal

Syntax
```
HORizontal:MODE:AUTO:LIMITrecordlen <NR1>
HORizontal:MODE:AUTO:LIMITrecordlen?
```

Arguments <NR1> is the record length limit in samples.

Examples HORIZONTAL:MODE:AUTO:LIMITRECORDLEN 1000 sets the record length limit to 1000 samples.

HORIZONTAL:MODE:AUTO:LIMITRECORDLEN? might return :HORIZONTAL:MODE:AUTO:LIMITRECORDLEN 1000000, indicating that the record length limit is set to 1,000,000 samples.

HORizontal:MODE:RECOrdlength

This command sets or queries the record length.

Group Horizontal

Syntax
```
HORizontal:MODE:RECOrdlength <NR1>
HORizontal:MODE:RECOrdlength?
```

Arguments <NR1> is the record length in samples. Manual mode lets you change the record length, while the record length is read only for Auto and Constant mode.

Examples HORIZONTAL:MODE:RECORDLENGTH 1000 sets the record length to 1000 samples.

HORIZONTAL:MODE:RECORDLENGTH? might return :HORIZONTAL:MODE:RECORDLENGTH 1000000, indicating that the record length is set to 1,000,000 samples

HORizontal:MODE:SAMPLERate

This command sets or queries the sample rate.

Group Horizontal

Syntax
```
HORizontal:MODE:SAMPLERate <NR1>
HORizontal:MODE:SAMPLERate?
```

Arguments <NR1> is the sample rate in samples per second.

Examples HORIZONTAL:MODE:SAMPLERATE 1e6 sets the sample rate to 1 million samples per second.

HORIZONTAL:MODE:SAMPLERATE? might return :HORIZONTAL:MODE:SAMPLERATE 5.0000E+6, indicating that the sample rate is set to 5 million samples per second.

HORizontal:MODE:SCAle

This command sets or queries the horizontal scale.

Group Horizontal

Syntax
```
HORizontal:MODE:SCAle <NR1>
HORizontal:MODE:SCAle?
```

Arguments <NR1> is the horizontal scale in seconds per division. The horizontal scale is read only for Manual mode.

Examples HORIZONTAL:MODE:SCALE 2e-9 sets the horizontal scale to 2 ns per division.

HORIZONTAL:MODE:SCALE? might return :HORIZONTAL:MODE:SCALE 20.0000E-6, indicating that the horizontal scale is set to 10 µs per division.

HORizontal:POSition

This command either positions the waveform horizontally on the display or returns its horizontal position on the display. When Horizontal Delay Mode is turned off, this command is equivalent to adjusting the HORIZONTAL POSITION knob on the front panel.

Group Horizontal

Syntax

```
HORizontal:POSition <NR3>
HORizontal:POSition?
```

Related Commands

HORizontal[:MAIn]:DELay:MODe

Arguments

<NR3> is from 0 to ≈100 and is the position of the trigger point on the screen (0 = left edge, 100 = right edge).

NOTE. *The upper limit of the trigger position is slightly limited by a value that is determined from the record length (upper limit = 100 - 1/record length).*

Examples

HORIZONTAL:POSITION 10 sets the trigger position of the waveform such that 10% of the display is to the left of the trigger position.

HORIZONTAL:POSITION? might return :HORIZONTAL:POSITION 5.0000E+01 indicates the trigger point displayed on the screen.

HORizontal:ROLL

This command sets or queries the Roll Mode status. Use Roll Mode when you want to view data at very slow sweep speeds. It is useful for observing data samples on the screen as they occur. This command is equivalent to selecting Horizontal/Acquisition Setup from the Horiz/Acq menu, selecting the Acquisition tab, and setting the Roll Mode to Auto or Off.

Group

Horizontal

Syntax

```
HORizontal:ROLL {AUTO|OFF|ON}
HORizontal:ROLL?
```

Arguments

AUTO enables Roll Mode, if the time/division is set appropriately.

OFF disables Roll Mode.

ON enables Roll Mode, if the time/division is set appropriately.

Examples

HORIZONTAL:ROLL ON enables Roll Mode.

HORIZONTAL:ROLL? might return :HORIZONTAL:ROLL OFF, indicating that the Roll Mode is disabled.

HORizontal:TIMEStamp:CH<x>? (Query Only)

This query returns the horizontal timebase for the channel. The channel is specified by x. The value of x can range from 1 through 4.

Group Horizontal

Syntax `HORizontal:TIMEStamp:CH<x>?`

Examples `HORIZONTAL:TIMESTAMP:CH1?` might return `:HORIZONTAL:TIMESTAMP:CH1 "timestamp not available"`, indicating that the timebase for the Channel 1 is not available.

HORizontal:TIMEStamp:REF<x>? (Query Only)

This query returns the horizontal timebase for the reference waveform. The reference is specified by x. The value of x can range from 1 through 4.

Group Horizontal

Syntax `HORizontal:TIMEStamp:REF<x>?`

Examples `HORIZONTAL:TIMESTAMP:REF1?` might return `:HORIZONTAL:TIMESTAMP:REF1 "timestamp not available"`, indicating that the timebase for the Reference 1 waveform is not available.

ID? (Query Only)

This query-only command returns identifying information about the instrument and related firmware.

Group Miscellaneous

Syntax `ID?`

Related Commands *IDN?

Examples `ID?` might return `:ID TEK/DPO7104,CF:91.1CT,FV:4.0.0`, indicating that the instrument model number is set to TEK/DPO7104, configured format is set to CF:91.1CT, and firmware version number is set to FV:4.0.0.

*IDN? (Query Only)

This query-only command returns the instrument identification code.

Group Miscellaneous

Syntax `*IDN?`

Related Commands ID?

Examples `*IDN?` might return `:*IDN TEKTRONIX,DPO7104,10000001,CF:91.1CT FV:01.00.912,`, indicating the instrument model number, serial number, configured number, and firmware version number.

IDNMultiscope? (Query Only)

This query returns instrument data on all MultiScope instruments. If the query is not run on the master instrument, data returned indicates that the instrument is running as a StandAlone Master. DPO70000SX Series only.

Group Miscellaneous

Syntax `IDNMultiscope?`

Examples `IDNMULTISCOPE??` might return `TEKTRONIX_MultiScope_[2],DPO73304SX_KR100017,DPO73304SX_KR100019,CF:
FV:7.1.91`.

IDNMultiscope:A? (Query Only)

This query returns the *IDN? response for the Master instrument. If the query is not run on the master instrument, data returned indicates that the instrument is running as a StandAlone Master. DPO70000SX Series only.

Group Miscellaneous

Syntax `IDNMultiscope:A?`

Examples `IDNMULTISCOPE:A??` might return `TEKTRONIX_Master,DPO73304SX,KR100017,CF:91.1CT FV:7.1.91`.

IDNMultiscope:B? (Query Only)

This query returns the *IDN? response for Extension 1. If the query is not run on the master instrument, data returned indicates that the instrument is running as a StandAlone Master. DPO70000SX Series only.

Group Miscellaneous

Syntax `IDNMultiscope:B?`

Examples `IDNMULTISCOPE:B??` might return `TEKTRONIX_Ext_1,DPO73304SX,KR100019,CF:91.1CT FV:7.1.91`.

IDNMultiscope:C? (Query Only)

This query returns the *IDN? response for Extension 2. If the query is not run on the master instrument, data returned indicates that the instrument is running as a StandAlone Master. DPO70000SX Series only.

Group Miscellaneous

Syntax `IDNMultiscope:C?`

Examples `IDNMULTISCOPE:C??` might return `TEKTRONIX_Ext_2,DPO73304SX,KR100020,CF:91.1CT FV:7.1.91`.

IDNMultiscope:D? (Query Only)

This query returns the *IDN? response for Extension 3. If the query is not run on the master instrument, data returned indicates that the instrument is running as a StandAlone Master. DPO70000SX Series only.

Group Miscellaneous

Syntax `IDNMultiscope:D?`

Examples `IDNMULTISCOPE:D??` might return `TEKTRONIX_Ext_3,DPO73304SX,KR100021,CF:91.1CT FV:7.1.91`.

LIMit? (Query Only)

This query-only command returns all settings for the Limit commands.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax `LIMit?`

Examples `LIMIT?` might return `:LIMIT:BEEP 0;COMPARE:CH1 NONE;CH2 NONE;CH3 NONE;CH4 NONE;REF1 NONE;REF2 NONE;REF3 NONE;REF4 NONE;MATH1 NONE;MATH2 NONE;MATH3 NONE;MATH4 NONE;:LIMIT:HARDCOPY 0;STATE 0;TEMPLATE:DESTINATION REF1;SOURCE CH2;TOLERANCE:HORIZONTAL 40.0000E-3;VERTICAL 40.0000E-3;:LIMIT:SRQ 0;STOPONVIOLATION 0;SAVEWFM 0;SAVEWFM:FILENAME "C:/TekScope/waveforms/";:LIMIT:LOG 0;EMAIL 0;HIGHLIGHTHITS 1;LOCK 1`.

LIMit:BEEP

This command causes the instrument to beep when the waveform data exceeds the limits set in the limit test (`LIMit:STATE` must be on).

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax
```
LIMit:BEEP {<NR1>|OFF|ON}
LIMit:BEEP?
```

Related Commands LIMit:STATE

Arguments <NR1> = 0 disables the beep; any other value enables the beep.

OFF disables the beep.

ON enables the beep.

Examples LIMIT:BEEPON causes a beep when any waveform data exceeds the limits specified in the limit test.

LIMIT:BEEP? might return :LIMIT:BEEP 0, indicating that the instrument will not beep when waveform data exceeds the limits specified in the limit test.

LIMit:COMpare (No Query Form)

This command resets the Limit Test comparison template.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax LIMit:COMpare {RESet}

Arguments RESet resets the Limit template to "None."

Examples LIMIT:COMPARE RESET sets the Limit template to None.

LIMit:COMpare:CH<x>

This command sets or queries the template against which to compare the waveform acquired from the channel specified by x. The template can be a waveform saved in any of the reference locations (REF1 through REF4) or none.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:COMpare:CH<x> {NONe|REF<x>}
LIMit:COMpare:CH<x>?
```

Arguments

NONe turns off template testing for the channel specified by CH<x>.

REF<x> selects which channel waveform to use as the template against which to compare the waveforms acquired by the specified channel (CH<x>).

Examples

LIMIT:COMPARE:CH1 REF1 specifies REF1 as the template waveform against which to compare waveforms acquired using Channel 1.

LIMIT:COMPARE:CH2? might return :LIMIT:COMPARE:CH2 REF4, indicating that the waveform acquired by Channel 2 is compared to the template waveform stored in REF4.

LIMit:COMpare:MATH<x>

This command sets or queries the template against which to compare the math waveform specified by x, which can be 1 through 4. The template can be a waveform saved in any of the reference locations (REF1 through REF4) or none.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group

Limit Test

Syntax

```
LIMit:COMpare:MATH<x> {NONe|REF<x>}
LIMit:COMpare:MATH<x>?
```

Arguments

NONe argument turns off template testing for the math waveform specified by MATH<x>.

REF<x> argument selects which reference waveform to use as the template against which to compare the math waveform specified by MATH<x>.

Examples

LIMIT:COMPARE:MATH1 REF1 specifies REF1 as the template waveform against which to compare the Math 1 waveform.

LIMIT:COMPARE:MATH2? might return :LIMIT:COMPARE:MATH2 REF4, indicating that the Math 2 waveform is compared to the template waveform stored in REF4.

LIMit:COMpare:REF<x>

This command sets or queries the template against which to compare the reference waveform specified by x, which can be 1 through 4. The template can be a waveform saved in any of the reference locations (REF1 through REF4) or none.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:COMpare:REF<x> {NONe|REF<x>}
LIMit:COMpare:REF<x>?
```

Arguments

NONe argument turns off template testing for the reference waveform specified by REF<x>.

REF<x> argument selects which reference waveform to use as the template against which to compare the reference waveform specified by REF<x>.

Examples

LIMIT:COMPARE:REF2 REF1 specifies REF2 as the template waveform against which to compare Reference 1.

LIMIT:COMPARE:REF2? might return :LIMIT:COMPARE:REF2 REF4, indicating that the Reference 4 waveform is compared to the template waveform stored in REF2.

LIMit:EMail

This command sets or queries whether an e-mail is generated when the source waveform data exceeds the limits specified for the limit test.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:EMail {<NR1>|OFF|ON}
LIMit:EMail?
```

Arguments

<NR1> = 0 disables the e-mail feature so that when the source waveform data exceeds the limits set by the limit test, an e-mail is not generated; any other value enables the e-mail feature so that when the source waveform data exceeds the

limits set by the limit test, an e-mail is generated. For queries, a 0 is returned if the e-mail feature is off; a 1 is returned if the e-mail feature is on.

OFF disables generating an e-mail when the source waveform exceeds the test limits.

ON enables generating an e-mail when the source waveform exceeds the test limits.

Examples

LIMIT:EMAIL1 enables the e-mail feature so that when the source waveform data exceeds the limits set by the limit test, an e-mail is generated.

LIMIT:EMAIL? might return :LIMIT:EMAIL 0, indicating that the e-mail feature is disabled and that an e-mail will not be generated when the source waveform exceeds the limits of the limit test.

LIMit:HARDCopy

This command sets or queries whether a hard copy operation is executed on the waveform when any waveform data exceeds the limit set in the limit test. LIMit:STATE must be set to ON for the hard copy operation to execute. The hard copy operation uses the port and prints in the format and layout specified by the HARDCopy commands.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group

Limit Test

Syntax

```
LIMit:HARDCopy {<NR1>|OFF|ON}
LIMit:HARDCopy?
```

Related Commands

HARDCopy:FILEName, HARDCopy:PORT, LIMit:STATE

Arguments

<NR1> = 0 disables the hard copy operation; any other value turns on the hard copy operation for the waveform when any waveform data exceeds the limits set by the limit test. For queries, a 0 is returned if the hard copy operation is off; a 1 is returned if the hard copy operation is on.

OFF disables the hard copy operation.

ON turns on the hard copy operation for the waveform when any waveform data exceeds the limits set by the limit test.

Examples

LIMIT:HARDCOPY ON specifies that the hard copy operation will occur for the waveform when any waveform data exceeds the limits specified in the limit test.

LIMIT:HARDCOPY? might return :LIMIT:HARDCOPY 0, indicating that the hard copy operation is disabled.

LIMit:HIGHLIGHTHits

This command sets or queries whether violation highlighting occurs when limit testing is active, and, if the RESET argument is set, clears the highlighting.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:HIGHLIGHTHits {<NR1>|OFF|ON}
LIMit:HIGHLIGHTHits?
```

Arguments

<NR1> = 0 disables the violation highlighting when limit testing is active; any other value enables the violation highlighting feature when limit testing is active. For queries, a 0 is returned if the violation highlighting feature is off; a 1 is returned if the violation highlighting feature is on.

OFF disables violation highlighting when limit testing is active.

ON enables violation highlighting when limit testing is active.

Examples

LIMIT:HIGHLIGHTHITS 1 enables the violation highlighting when the limit testing is active.

LIMIT:HIGHLIGHTHITS? might return :LIMIT:HIGHLIGHTHITS 0, indicating that the violation highlighting feature is disabled.

LIMit:HIGHLIGHTHits:RESet (No Query Form)

This command resets the hits highlighting for limit testing.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:HIGHLIGHTHits:RESet
```

Examples `LIMIT:HIGHLIGHTHITS:RESET` resets the violation highlighting displayed while limit testing.

LIMit:LOCk

This command sets or queries whether vertical scaling and positioning affect both source and template for template comparison pairs.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax
```
LIMit:LOCk {<NR1>|OFF|ON}
LIMit:LOCk?
```

Arguments `<NR1>` = 0 disables the lock feature; any other value enables the lock feature. For queries, a 0 is returned if the lock feature is off; a 1 is returned if the lock feature is on.

`OFF` disables the lock feature.

`ON` enables the lock feature.

Examples `LIMIT:LOCK ON` enables the lock feature so that vertical scaling and positioning affect both the source and the template for template comparison pairs.

`LIMIT:LOCK?` might return `:LIMIT:LOCK 0`, indicating that the lock feature is disabled.

LIMit:LOG

This command sets or queries whether a log file is saved when the source waveform data exceeds the test limits.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax
```
LIMit:LOG {<NR1>|OFF|ON}
LIMit:LOG?
```

Arguments

<NR1> = 0 disables the save log file feature so that when the source waveform data exceeds the limits set by the limit test, a log file is not saved; any other value enables the save log file feature so that when the source waveform data exceeds the limits set by the limit test, a log file is saved. For queries, a 0 is returned if the save log file feature is off; a 1 is returned if the save log file feature is on.

OFF disables saving a log file when the source waveform exceeds the test limits.

ON enables saving a log file when the source waveform exceeds the test limits.

Examples

LIMIT:LOG 1 enables the save log file feature so that when the source waveform data exceeds the limits set by the limit test, a log file is saved.

LIMIT:LOG? might return :LIMIT:LOG 0, indicating that the save log file feature is disabled and that a log file will not be saved when the source waveform exceeds the limits of the limit test.

LIMit:SAVEWFM

This command sets or queries whether the source waveform is saved when the source waveform data exceeds the test limits.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group

Limit Test

Syntax

```
LIMit:SAVEWFM {<NR1>|OFF|ON}
LIMit:SAVEWFM?
```

Arguments

<NR1> = 0 disables the save waveform feature so that when the source waveform data exceeds the limits set by the limit test, the source waveform is not saved; any other value enables the save waveform feature so that when the source waveform data exceeds the limits set by the limit test, the source waveform is saved. For queries, a 0 is returned if the save waveform feature is off; a 1 is returned if the save waveform feature is on.

OFF disables saving the source waveform when it exceeds the test limits.

ON enables saving the source waveform when it exceeds the test limits.

Examples

LIMIT:SAVEWFM 1 enables the save waveform feature so that when the source waveform data exceeds the limits set by the limit test, the source waveform is saved.

LIMIT:SAVEWFM? might return :LIMIT:SAVEWFM 0, indicating that the save waveform feature is disabled and that the source waveform will not be saved when it exceeds the limits of the limit test.

LIMit:SAVEWFM:FILEName

This command sets or queries the path where waveforms or log files will be saved when the waveform data exceeds the limits set by the limit test.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:SAVEWFM:FILEName <QString>
LIMit:SAVEWFM:FILEName?
```

Arguments <QString> argument is a string containing the path of where the waveform will be saved.

Examples LIMIT:SAVEWFM:FILENAME "C:/TekScope/waveforms" sets the path where a waveform will be saved when it exceeds the limits of the limit test.

LIMIT:SAVEWFM:FILENAME? might return :LIMIT:SAVEWFM:FILENAME "C:/TekScope/waveforms", indicating the path were the waveform will be saved when it exceeds the limits of the limit test.

LIMit:SRQ

This command sets or queries whether a Service Request Interrupt (SRQ) is generated when the waveform data falls outside of the test limits.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:SRQ {<NR1>|OFF|ON}
LIMit:SRQ?
```

Arguments

<NR1> = 0 disables sending an SRQ when the waveform data falls outside of the limits set by the limit test; any other value enables generation of an SRQ when the waveform data falls outside of the limits set by the limit test. For queries, a 0 is returned if SRQ is off; a 1 is returned if the SRQ is on.

OFF disables generation of an SRQ when the waveform data falls outside of the limits set by the limit test.

ON enables generation of an SRQ when the waveform data falls outside of the limits set by the limit test. If an SRQ is generated, the instrument sends the status event "Limit testing failed."

Examples

LIMIT:SRQ 1 specifies that an SRQ will be generated when the waveform data falls outside of the limits set by the limit test. If an SRQ is generated, an status event of "Limit testing failed" is output.

LIMIT:SRQ? might return :LIMIT:SRQ 0, indicating that an SRQ will not be generated when the waveform data falls outside of the limits set by the limit test.

LIMit:STATE

This command sets limit testing on or off or queries whether limit testing is in effect.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group

Limit Test

Syntax

```
LIMit:STATE {<NR1>|OFF|ON}
LIMit:STATE?
```

Arguments

<NR1> = 0 disables the limit testing operation; any other value turns on limit testing of waveforms. For queries, a 0 is returned if limit testing is off; a 1 is returned if the limit testing is on.

OFF disables limit testing.

ON turns on limit testing of waveforms.

Examples

LIMIT:STATE ON specifies that limit testing of waveforms is on.

LIMIT:STATE? might return :LIMIT:STATE 0, indicating that the limit testing of waveforms is off.

LIMit:STATus? (Query Only)

This command queries the state of limit testing.

NOTE. The command works only if the LT Limit Test option is enabled.

Group Limit Test

Syntax

```
LIMit:STATus?
```

Examples `LIMIT:STATUS?` might return `:LIMIT:STATUS PASS`, indicating that the limit test is passing.

LIMit:STOPOnviolation

This command sets or queries whether acquisitions are stopped when the waveform data exceeds the test limits.

NOTE. The command works only if the LT Limit Test option is enabled.

Group Limit Test

Syntax

```
LIMit:STOPOnviolation {<NR1>|OFF|ON}
LIMit:STOPOnviolation?
```

Arguments `<NR1>` = 0 disables the stop on violation feature; any other value enables the stop on violation feature so that when the waveform data exceeds the limits set by the limit test, acquisitions are stopped. For queries, a 0 is returned if the stop on violation feature is off; a 1 is returned if the stop on violation feature is on.

`OFF` disables the stop on violation feature.

`ON` enables the stop on violation feature so that when the waveform data exceeds the limits set by the limit test, acquisitions are stopped.

Examples `LIMIT:STOPONVIOLATION 1` enables the stop on violation feature so that when the waveform data exceeds the limits set by the limit test, acquisitions are stopped.

`LIMIT:STOPONVIOLATION?` might return `:LIMIT:STOPONVIOLATION 0`, indicating that the stop on violation feature is disabled.

LIMit:TEMPlate:DESTination

This command sets or queries the destination reference waveform that the LIMit:TEMPlate:STORe command will use.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:TEMPlate:DESTination {REF1|REF2|REF3|REF4}
LIMit:TEMPlate:DESTination?
```

Arguments `REF<x>` specifies the reference waveform destination in which the template waveform is to be stored.

Examples `LIMIT:TEMPLATE:DESTINATION REF2` specifies that the template waveform referred to with the LIMit:TEMPlate:STORe command is stored as the REF2 waveform.

`LIMIT:TEMPLATE:DESTINATION?` might return `:LIMIT:TEMPLATE:DESTINATION REF3`, indicating the REF3 waveform is used to store the template waveform referred to with the LIMit:TEMPlate:STORe command.

LIMit:TEMPlate:SOUrce

This command sets or queries the channel, math waveform, or reference waveform that the LIMit:TEMPlate:STORe command will use.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group Limit Test

Syntax

```
LIMit:TEMPlate:SOUrce
{CH1|CH2|CH3|CH4|MATH1|MATH2|MATH3|MATH4|REF1|REF2|REF3|REF4}
LIMit:TEMPlate:SOUrce?
```

Arguments `CH1–CH4` specifies the channel used by the LIMit:TEMPlate:STORe command.

`MATH1–MATH4` specifies the math waveform used by the LIMit:TEMPlate:STORe command.

REF1–REF4 specifies the reference waveform used by the LIMit:TEMPlate:STORe command.

Examples

LIMIT:TEMPLATE:SOURCE CH2 specifies that the template waveform for limit tests is the waveform currently acquired using Channel 2.

LIMIT:TEMPLATE:SOURCE? might return :LIMIT:TEMPLATE:SOURCE MATH3, indicating that the template waveform for limit tests is the waveform currently stored as the Math 3 waveform.

LIMit:TEMPlate:STORe (No Query Form)

This command (no query form) saves the specified source waveform to the specified reference or file name.

NOTE. *The command works only if the LT Limit Test option is enabled.*

Group

Limit Test

Syntax

LIMit:TEMPlate:STORe <wfm>,{REF<x>|<file path>}

Arguments

<wfm> specifies the waveform that will be saved as the template. The source of the waveform can be CH<x> (where x is 1 through 4 for four channel instruments), MATH<x>, or Ref<x> (where x is 1 through 4).

REF<x> specifies a reference location in which the template waveform will be stored. The reference waveform specified by x, which can be 1 through 4.

<file path> specifies a file path where the template waveform will be stored. This argument is a quoted string.

Examples

LIMIT:TEMPLATE:STORE CH3,REF1 creates a template reference waveform, REF1, using Channel 3 as its source.

LIMIT:TEMPLATE:STORE CH1,"TEMPLATE.WFM" creates a template reference waveform, template.wfm, using Channel 1 as the source.

LIMit:TEMPlate:TOLerance:HORizontal

This command sets or queries the amount in units of horizontal divisions, by which the source waveform is varied horizontally when creating the destination waveform.

NOTE. The command works only if the LT Limit Test option is enabled.

Group Limit Test

Syntax
```
LIMit:TEMPlate:TOLerance:HORizontal <NR3>
LIMit:TEMPlate:TOLerance:HORizontal?
```

Arguments <NR3> is the amount in horizontal divisions, by which the current source waveform is allowed to deviate from the template waveform without exceeding the limits set in the limit test. The range is 0 to 5 divisions.

Examples `LIMIT:TEMPLATE:TOLERANCE:HORIZONTAL 1.0` specifies that the current waveform is close enough to the template waveform if it is within ±1.0 horizontal division.

`LIMIT:TEMPLATE:TOLERANCE:HORIZONTAL?` might return `:LIMIT:TEMPLATE:TOLERANCE:HORIZONTAL 2.0`, indicating that the current waveform is close enough to the template waveform if it is within ±2.0 horizontal divisions.

LIMit:TEMPlate:TOLerance:VERTical

This command sets or queries the amount in units of vertical divisions, by which the source waveform is varied vertically when creating the destination waveform.

NOTE. The command works only if the LT Limit Test option is enabled.

Group Limit Test

Syntax
```
LIMit:TEMPlate:TOLerance:VERTical <NR3>
LIMit:TEMPlate:TOLerance:VERTical?
```

Arguments <NR3> is the amount in vertical divisions, by which the current source waveform is allowed to deviate from the template waveform without exceeding the limits set in the limit test. The range is 0 to 5 divisions.

Examples `LIMIT:TEMPLATE:TOLERANCE:VERTICAL 1.0` specifies that the current waveform is close enough to the template waveform if it is within ±1.0 vertical division.

LIMIT:TEMPLATE:TOLERANCE:VERTICAL? might return :LIMIT:TEMPLATE:TOLERANCE:VERTICAL 2.0, indicating that the current waveform is close enough to the template waveform if it is within ±2.0 vertical divisions.

LINKTRaining:ACQTime

This 70KSX command sets the maximum Acquire Time for Link Training from 1 to 10 seconds. The default value is 2 seconds. The link training process is defined by the IEEE to complete within 500 ms. So, the default 2 seconds is sufficient for most purposes. However, some devices can be put into a mode where the Link Training process is repeated continuously. In this special case you may want to increase the acquire time. Increasing the acquire time will capture more rows of protocol data into the Link Training Results Table. Naturally, you will have to wait a little longer for the acquisition to complete.

The Link Training application is only available on the 70KSX oscilloscopes.

Group Link Training

Syntax

```
LINKTRaining:ACQTime <NR1>
LINKTRaining:ACQTime?
```

Related Commands LINKTRaining:STANdard, LINKTRaining:LANE1, LINKTRaining:LANE2, LINKTRaining:LANE3, LINKTRaining:LANE4, LINKTRaining:MARK, LINKTRaining:DECOde, LINKTRaining:TRIGgeron, LINKTRaining:ARMscope

Arguments <NR1> is the maximum Acquire Time for Link Training from 1 to 10 seconds.

Examples LINKTRAINING:ACQTIME 5 sets the Link Training acquire time to 5 (seconds).

LINKTRAINING:ACQTIME 10 sets the Link Training acquire time to 10 (seconds).

LINKTRAINING:ACQTIME? might return the 0 to 10 (seconds) acquire time setting.

LINKTRaining:ARMscope

This 70KSX command sets or queries the ARMscope setting of the Link Training application on the 70KSX family of scopes. The default argument value is OFF. To use the Link Training application, the State argument must first be set to ON. With the DUT Lane “powered down” (flat-line), you then Arm the scope

for single-step capture using this command. When you subsequently initiate Link Training on the DUT ("power-up" the lane), the Control Channel data and digitized waveforms are captured by the scope.

At a minimum, to capture the Link Training data into a Results Table, both the LINKTTRaining:STAte ON and LINKTRaining:ARMscope ON commands must be issued, in that order. When Link Training is initiated on the DUT lane, the Results Table data and digitized waveforms will be captured. You may select other Link Training settings using the related commands shown below to control those features.

Initiating Link Training on the DUT is a DUT-dependent operation that is usually accomplished by a (serial or other bus) command to the DUT to power-up the Lane under test. Sometimes, it can be as simple as un-plugging and re-plugging the signal connector for the lane.

Note that it is not necessary to set the ARMscope to OFF as this is done automatically by the system.

To summarize, the minimum sequence of events to capture 100G Ethernet KR4 from Default Setup is
:LINKTRaining:STATE ON
:LINKTRaining:ARMscope ON
Initiate Link Training on the Lane under test on the DUT.
The Results Table and Waveforms are captured on the scope.

To repeat the experiment, all you have to do is power-down the lane on the DUT, re-issue the ARMscope command, and power-up the lane on the DUT again.

Link Training is only available on the 70KSX oscilloscopes.

Group Link Training

Syntax

```
LINKTRaining:ARMscope <OFF | ON>
LINKTRaining:ARMscope?
```

Arguments

`OFF` indicates ARMscope is OFF.

`ON` indicates ARMscope is ON.

Examples

`LINKTRAINING:ARMSCOPEON` sets Link Training ARMscope to ON.

`LINKTRAINING:ARMSCOPE`? might return the ARMscope as ON.

LINKTRaining:DECOde

This 70KSX command sets or queries the Decode setting of the Link Training application. The default argument value is ON, which decodes the Control Channel information on the waveform display. Other system decode parameters optionally allow the Control Channel Hex Decode to be expanded to also include bit-level decode. The only plausible reason to turn off decoding is to save time when you are capturing a very long record using TRIGgeron FIRst_frame or LASt_frame. With TRIGgeron ALL_frames, the overhead for decoding is very small.

Link Training is only available on the 70KSX family of oscilloscopes.

Group Link Training

Syntax

```
LINKTRaining:DECOde <OFF | ON>
LINKTRaining:DECOde?
```

Arguments

`OFF` indicates decode is OFF.

`ON` indicates decode is ON.

Examples

`LINKTRAINING:DECODE ON` sets Link Training Decode to ON.

`LINKTRAINING:DECODE OFF` sets Link Training Decode to OFF.

`LINKTRAINING:DECODE?` may return the value ON.

LINKTRaining:EQUalization

This 70KSX command sets the equalization value for Link Training. The default setting is zero. The units are dB. If you are probing a signal near to the transmitter, you likely do not need to increase the equalization value. However, if you are probing on a backplane, or other situation where the signal is seriously attenuated, you may want to increase the equalization value.

Note that this equalization only applies to the trigger path in the oscilloscope. It does not apply to the data path. Use the SDLA option to provide equalization/de-embedding on the data path. Increasing the trigger path equalization value will improve the protocol data recovered into the Link Training Results Table when you are probing a seriously attenuated signal. Use the SDLA option to improve the equalization of the data path waveforms displayed on the oscilloscope.

Since the protocol data (Ethernet Control Channel Coefficient Updates and Status Responses) is transmitted at ¼ the nominal bit rate (25 Gb/s for 100G Ethernet),

the equalization value should be set considering ½ that reduced bitrate. For example, for 100G Ethernet, the protocol data is transmitted at 6.25 Gb/s, so set the equalization value considering half of that rate, 3.125 Gb/s.

The Link Training application is only available on the 70KSX scopes.

Group Link Training

Syntax

```
LINKTRaining:EQUalization <NR1>
LINKTRaining:EQUalization?
```

Related Commands LINKTRaining:LANE1, LINKTRaining:LANE2, LINKTRaining:LANE3, LINKTRaining:LANE4, LINKTRaining:MARK, LINKTRaining:DECOde, LINKTRaining:TRIGgeron, LINKTRaining:STANdard

Arguments <NR1> is the equalization (0 to 15) in dB.

Examples `LINKTRAINING:EQUALIZATION`: 5 sets Link Training equalization to 5 (dB).

`LINKTRAINING:EQUALIZATION` 15 sets Link Training equalization to 15 (dB).

`LINKTRAINING:EQUALIZATION`? might return the 0 to15 (dB) equalization setting.

LINKTRaining:LANE

This command sets or queries the activation of the individual communications lanes under test. The defaults are Lane1 ON, and Lane2 OFF. It is taken as given that Lane1 uses oscilloscope Ch1 and Ch3, Lane2 uses oscilloscope Ch2 and Ch4 (to maximize sample rate). ATI DPO70000SX oscilloscopes only support Lane1 (on Ch1 and Ch3). <X> is the lane number and can be 1 or 2.

One or two lanes may be tested on a single 4-channel DPO70000SX oscilloscope. To test up to 4 lanes simultaneously, two oscilloscopes running in Standalone mode may be used. This is necessary because the Time-Sync Standalone mode does not support independent triggering by multiple oscilloscopes. The associated Link Training Arm PI command or UI button, may be used to arm each oscilloscope individually. When the Link Training signal is generated by the devices under test, the individual oscilloscopes will each trigger, capturing the waveforms on their respective displays and the protocol information in their respective Results Tables.

Note that each Lane requires two oscilloscope channels, because the lanes are bi-directional (Full Duplex).

Link Training is available on DPO70000SX oscilloscopes only.

Group Link Training

Syntax

```
LINKTRaining:LANE<x> {<NR1>|OFF|ON}
LINKTRaining:LANE?
```

Related Commands LINKTRaining:STAte, LINKTRaining:MARK, LINKTRaining:DECOde, LINKTRaining:TRIGgeron, LINKTRaining:ARMscope, LINKTRaining:STANdard

Arguments <NR1> = 0 disables the specified lane; any other value enables the specified lane.

OFF disables the specified lane.

ON enables the specified lane.

Examples LINKTRAINING:LANE1 ON sets Link Training Lane1 to ON

LINKTRAINING:LANE1 OFF sets Link Training Lane1 to OFF

LINKTRAINING:LANE2 ON sets Link Training Lane2 to ON

LINKTRAINING:LANE2 OFF sets Link Training Lane2 to OFF

LINKTRAINING:LANE1? may return :LINKTRAINING:LANE1 ON indicating LANe1 is ON.

LINKTRAINING:LANE2? may return :LINKTRAINING:LANE2 OFF indicating LANE2 is OFF.

LINKTRaining:MARK

This command sets or queries the Mark setting of the Link Training application. The default argument value is FRAme, which marks the Frame Marker. When the argument is set to CONtrol, the frame Control Channel information is marked (32 cells of differential-Manchester-encoded request/response data). When the argument is TRAining, the PRBS Training Data is marked.

Marking the PRBS Training Data is useful for giving you the beginning/ending of the Training Data section of the frame when you want to use cursor-gated DPOJET measurements.

Link Training is only available on the 70KSX family of oscilloscopes.

Group Link Training

Syntax `LINKTRaining:MARK <FRAme | CONtrol | TRAining>`

Related Commands LINKTRaining:STAte, LINKTRaining:LANE1, LINKTRaining:LANE2, LINKTRaining:LANE3, LINKTRaining:LANE4, LINKTRaining:MARK, LINKTRaining:DECOde, LINKTRaining:TRIGgeron, LINKTRaining:ARMscope, LINKTRaining:STANdard

Arguments `FRAme` sets Link Training MARK to FRAME.

`CONtrol` sets Link Training MARK to CONTROL.

`TRAining` sets Link Training MARK to TRAINING.

Examples `LINKTRAINING:MARK` sets Link Training MARK to FRAME.

`LINKTRAINING:MARK` sets Link Training MARK to CONTROL.

`LINKTRAINING:MARK` sets Link Training MARK to TRAINING.

`LINKTRAINING:MARK?` may return the value FRAME.

LINKTRaining:SETUP (No Query Form)

This 70KSX command saves or restores custom Link Training setup files. The setup files are complete instrument setup files. The setup files are saved/restored depending on the current values for Link Training Standard, Trigger-On, and Lane selections. You only need to use this command when you desire to preserve custom Vertical, Horizontal, or Zoom settings for the particular Link Training mode you are using.

When Link Training is enabled, you may dismiss the pop-up control window and make custom changes to the Vertical, Horizontal, or Zoom settings, for example, to accommodate various probes or record length requirements. If you want to save these settings for future use, just issue the LINKTRaining:SETUP SAVe command. You can restore the factory Link Training setup at any time by using the LINKTRaining:SETUP RESTore command. UI buttons for these operations are also supplied via the Customize button on the pop-up UI for Link Training.

You do not need to enter any file or directory names to use these commands. When you save your custom setup with this command, the Link Training setup file pertaining to the current Standard, Trigger-On, and Lane selections is overwritten with your new setup. You may restore the factory Link Training setup file at any time.

Again, these commands save/restore the complete instrument setup for the particular Link Training mode currently selected.

The Link Training application is only available on the 70KSX scopes.

Group Link Training

Syntax `LINKTRaining:SETUP <SAVe |RESTore>`

Related Commands LINKTRaining:STANdard, LINKTRaining:LANE, LINKTRaining:TRIGgeron, LINKTRaining:ARMscope

Arguments `SAVe` saves your custom Link Training setup.

`RESTore` restores the custom Link Training setup.

Examples `LINKTRAINING:SETUP SAVe` saves the setup file per the current Standard, Trigger-On, and Lane selections.

`:LINKTRaining:SETUP RESTore` sets Link Training State to OFF.

LINKTRaining:STANdard

This DPO70000SX command sets or queries the communication standard for the Link Training application on DPO70000SX oscilloscopes. The default argument value is Ethernet_100G_KR4. Link Training is only available on the DPO70000SX family of oscilloscopes.

Group Link Training

Syntax
```
LINKTRaining:STANdard <Ethernet_100G_KR4 |
Ethernet_100G_CR4>
LINKTRaining:STANdard?
```

Related Commands LINKTRaining:STAte, LINKTRaining:LANE1, LINKTRaining:LANE2, LINKTRaining:LANE3, LINKTRaining:LANE4, LINKTRaining:MARK, LINKTRaining:DECOde, LINKTRaining:TRIGgeron, LINKTRaining:ARMscope

Arguments `Ethernet_100G_KR4` sets Link Training Standard to Ethernet_100G_KR4.

`Ethernet_100G_CR4` sets Link Training Standard to Ethernet_100G_CR4.

Examples `LINKTRAINING:STANDARD Ethernet_100G_KR4` sets Link Training Standard as Ethernet_100G_KR4.

`LINKTRAINING:STANDARD?` may return the Standard as Ethernet_100G_KR4.

LINKTRaining:STAte

This DPO70000SX command sets or queries the state of the Link Training application on the DPO70000SX family of oscilloscopes. The default argument value is OFF. To use the Link Training application, the State argument must first be set to ON.

At a minimum, to capture the Link Training data into a Results Table, the LINKTRaining:ARMscope ON command also must be issued after the Link Training State is ON. Once the State is ON and the ARMscope is ON, the Link Training Results Table data will be captured when Link Training is initiated on the DUT.

Initiating Link Training on the DUT is a DUT-dependent operation that is usually accomplished by a (serial or other bus) command to the DUT to power-up the Lane under test. Sometimes, it can be as simple as un-plugging and re-plugging the signal connector for the lane.

To summarize, the minimum sequence of events to capture 100G Ethernet KR4 from Default Setup is
:LINKTRaining:STATE ON
:LINKTRaining:ARMscope ON
Initiate Link Training on the Lane under test on the DUT.
The Results Table and Waveforms are captured on the scope.

Link Training is only available on the DPO70000SX oscilloscopes.

Group Link Training

Syntax

```
LINKTRaining:STAte <OFF | ON>
LINKTRaining:STAte?
```

Related Commands LINKTRaining:STANdard, LINKTRaining:LANE1, LINKTRaining:LANE2, LINKTRaining:LANE3, LINKTRaining:LANE4, LINKTRaining:MARK, LINKTRaining:DECOde, LINKTRaining:TRIGgeron, LINKTRaining:ARMscope

Arguments

`OFF` sets Link Training State to OFF.

`ON` sets Link Training State to ON.

Examples

`LINKTRAINING:STATE` ON sets Link Training State to ON.

`LINKTRAINING:STATE` OFF sets Link Training State to OFF.

`LINKTRAINING:STATE` might return the State as ON.

LINKTRaining:TRIGgeron

This DPO70000SX command sets or queries the TriggerOn setting of the Link Training application. The default argument value is ALL_FRAMES, which attempts to trigger the scope whenever any Frame Control Channel contains new (changed) data values to capture FastFrame record waveforms.

When the TriggerOn argument is FIRST_FRAME or LAST_FRAME, a single record waveform is captured, and the user may select the record length and horizontal position, among other settings, for this waveform.

When the TriggerOn argument is ALL_FRAMES, FastFrame records are captured containing the decoded and marked Control Channel information. The user may select the FastFrame Length and maximum number of Frames (< 512 at this writing).

Note that the Results Table captures all of the Control Channel data on both scope channels of a given lane, no matter what the TriggerOn argument value is chosen, because the data is captured by hardware in real-time (“100% live”). This means that the data in the Results Table is derived directly from the signal, not from the digitized waveform (that would be decoding, and we do that where possible).

However, only some of the Control Channel values in the Results Table will be backed with Waveforms on the display, because the Frames may arrive faster than the Acquisition System can capture and store the FastFrame waveforms, or because the Link Training process takes more time, up to 500ms, than can be captured in a single long record at any reasonable sample rate. When rows in the Results Table are not backed by an acquired waveform on the display, three asterisks (***) are shown in the Frame column.

Remember, all of the Control Channel data is captured in the Results Table for all three Triggeron settings. So, you should rely on the data in the Results Table, using the waveform decoding as confirmation.

Don’t forget that you can use cursor-gated DPOJET measurements to analyze the marked waveforms that are captured. The cursor positioning is made easier if you set the Link Training Marks to be TRAining to delineate the PRBS Training Data section of the Frames. Then you can use the extents of the MARK to better position the cursors for the measurements.

Link Training is only available on the DPO70000SX family of oscilloscopes.

Group Link Training

Syntax

```
LINKTRaining:TRIGgeron <FIRst_frame | LASt_frame |
ALL_frames>
```

Arguments

FIRst_frame sets Link Training TRIGgeron to FIRst_frame

LASt_frame sets Link Training TRIGgeron to LASt_frame

ALL_frames sets Link Training TRIGgeron to ALL_frames

Examples

LINKTRAINING:TRIGGERON ALL_frames sets Link Training TRIGgeron to ALL_frames.

LINKTRAINING:TRIGGERON FIRst_frame sets Link Training TRIGgeron to FIRst_frame.

LINKTRAINING:TRIGGERON LASt_frame Ssets Link Training TRIGgeron to LASt_frame.

LINKTRAINING:TRIGGERON? may return the values FIRst_frame, LASt_frame, or ALL_frames.

LOCk

This command enables or disables all front panel buttons and knobs, including the touch screen. There is no front panel equivalent.

Group

Miscellaneous

Syntax

```
LOCk {ALL|NONe}
LOCk?
```

Related Commands

UNLock

Arguments

ALL disables all front panel controls.

NONe enables all front panel controls. This is equivalent to the UNLock ALL command.

If the instrument is in the Remote With Lockout State (RWLS), the LOCk NONe command has no effect. For more information, see the ANSI/IEEE Std 488.1-1987 Standard Digital Interface for Programmable Instrumentation, section 2.8.3 on RL State Descriptions.

Examples

LOCK ALL locks the front panel controls.

LOCK? might return :LOCK NONE, indicating that the front panel controls are enabled by this command.

*LRN? (Query Only)

This query-only command returns the commands that list the instrument settings (except for configuration information for the calibration values), allowing you to record or "learn" the current instrument settings. You can use these commands to return the instrument to the state it was in when you made the *LRN? query. This command is identical to the SET? Command.

Group Miscellaneous

Syntax
```
*LRN?
```

Related Commands SET?

Examples *LRN? might return the following response:
```
:ACQUIRE:STOPAFTER RUNSTOP;STATE 1;MODE SAMPLE;NUMENV 10;NUMAVG
16;NUMSAMPLES 16000;SAMPLINGMODE IT;:FASTACQ:STATE
0;:APPLICATION:GPKNOB1:ACTIVE 0;:APPLICATION:GPKNOB2:ACTIVE
0;:APPLICATION:WINDOW:HEIGHT 236;WIDTH
640;:APPLICATION:SCOPEAPP:STATE RUNNING;WINDOW
FULLSCREEN;:APPLICATION:EXTAPP:STATE
NOTRUNNING;:VARIABLE:EVENT:REPORT BOTH;:AUXOUT:SOURCE
ATRIGGER;EDGE FALLING;:CMDBATCH 1;:HEADER 1;:LOCK
NONE;:ROSC:SOURCE INTERNAL;:VERBOSE 1;:ALIAS:STATE
0;:DISPLAY:CLOCK 1;COLOR:PALETTE:IMAGEVIEW
TEMPERATURE;RECORDVIEW NORMAL;USER:GRATICULE 165,50,15;CH1
180,50,100;CH2 300,50,100;CH3 60,50,100;CH4 240,50,100;REF1
0,90,0;REF2 0,90,100;REF3 60,90,100;REF4 240,90,100;MATH1
160,50,100;MATH2 40,60,100;MATH3 120,60,100;MATH4
195,50,100;HISTOGRAM 320,50,100;CARET 150,50,100;MASK
0,25,75;MASKHIGHLIGHT 140,50,100;:DISPLAY:COLOR:MATHCOLOR
DEFAULT.
```

MARK

This commands moves to the next or previous reference mark, or it queries the current mark setup.

Group Search and 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.

Examples MARK NEXT moves to the next reference mark on the right.

MARK? might return :MARK:SELECTED:STATE 1;START 64.4000;FOCUS 64.4000;ZOOM:POSITION 64.4000;SCALE -3.1416;:MARK:SELECTED:END 64.4000;SOURCE CH1;OWNER SEARCH1;MARKSINCOLUMN 1;LABEL "+Edge: level=420.32mV";:MARK:FREE 1011;TOTAL 13, which is the current mark setup.

MARK:CREATE (No Query Form)

This command creates a mark on a specified waveform or all waveforms in a column.

Group Search and Mark

Syntax MARK:CREATE {CH<x>|MATH<x>|REF<x>|COLUMN}

Arguments CH<x> creates the mark on a channel waveform, where <x> is the channel number and can be 1, 2, 3, or 4.

MATH<x> creates the mark on the math waveform, where <x> is the channel number and can be 1, 2, 3, or 4.

REF<x> creates the mark on a reference waveform, where <x> is the reference waveform number can be 1, 2, 3, or 4.

COLUMN creates marks on all waveforms in the current zoom pixel column.

Examples MARK:CREATE CH1 creates a mark on Channel 1.

MARK:DELEte (No Query Form)

This command deletes a mark on a particular waveform, all waveforms in a column, the selected mark, or all marks.

Group Search and Mark

Syntax MARK:DELEte {CH<x>|MATH<x>|SELECTED|REF<x>|ALL|COLUMN}

Arguments CH<x> deletes the mark on a channel waveform, where <x> is the channel number and can be 1, 2, 3, or 4.

MATH<x> deletes the mark on the math waveform, where <x> is the channel number and can be 1, 2, 3, or 4.

SELECTED deletes the mark on the selected waveform.

REF<x> deletes the mark on a reference waveform, where <x> is the reference waveform number and can be 1, 2, 3, or 4.

ALL deletes all marks on all waveforms.

COLUMN deletes marks on all waveforms in the current zoom pixel column.

Examples MARK:DELETE CH1 deletes the mark on the Channel 1 waveform.

MARK:FREE? (Query Only)

This query-only command returns how many user marks are available for use.

Group Search and Mark

Syntax MARK:FREE?

Examples MARK:FREE? might return :MARK:FREE 1011, indicating there are 1,011 user marks available for use.

MARK:SELECTED:END? (Query Only)

This query-only command returns the end of the selected mark, 0 to 100% of the waveform.

Group Search and Mark

Syntax MARK:SELECTED:END?

Examples MARK:SELECTED:END? might return :MARK:SELECTED:END 93.0000, indicating that the selected mark is at 93% of the waveform.

MARK:SELECTED:FOCUS? (Query Only)

This query-only command returns the focus of the selected mark, 0 to 100% of the waveform.

Group Search and Mark

Syntax `MARK:SELECTED:FOCUS?`

Examples `MARK:SELECTED:FOCUS?` might return `:MARK:SELECTED:FOCUS 50.0000`, indicating that the selected mark is at 50% of the waveform.

MARK:SELECTED:LABel

This command sets or queries the label for the selected mark.

Group Search and Mark

Syntax
```
MARK:SELECTED:LABel <QString>
MARK:SELECTED:LABel?
```

Arguments `<QString>` is the quoted string label for the mark.

Examples `MARK:SELECTED:LABEL “Test 42.3”` sets the selected mark to a User mark and changes the mark description to Test 42.3.

`MARK:SELECTED:LABEL?` might return `:MARK:SELECTED:LABEL "+Edge: level=420.32mV"`, indicating that the label description for the selected mark is +Edge: level=420.32 mV.

MARK:SELECTED:MARKSINCOLumn? (Query Only)

This query-only command returns the number of marks in the current zoom pixel column.

Group Search and Mark

Syntax `MARK:SELECTED:MARKSINCOLumn?`

Arguments `MARK:SELECTED:MARKSINCOLUMN?` might return `:MARK:SELECTED:MARKSINCOLUMN 1`, indicating there is 1 mark in the pixel column.

MARK:SELECTED:OWNer? (Query Only)

This query-only command returns the owner of the selected mark.

Group Search and Mark

Syntax `MARK:SELECTED:OWNer?`

Examples `MARK:SELECTED:OWNER?` might return: `:MARK:SELECTED:OWNER SEARCH1`, indicating that the selected mark is a Search1 mark.

MARK:SELECTED:SOUrce? (Query Only)

This query-only command returns the source waveform for the selected mark.

Group Search and Mark

Syntax `MARK:SELECTED:SOUrce?`

Examples `MARK:SELECTED:SOURCE?` might return `:MARK:SELECTED:SOURCE CH2`, indicating that Channel 2 is the selected mark source.

MARK:SELECTED:STARt? (Query Only)

This query-only command returns the starting point of the selected mark, 0 to 100% of the waveform.

Group Search and Mark

Syntax `MARK:SELECTED:STARt?`

Examples `MARK:SELECTED:START?` might return `:MARK:SELECTED:START 5.8000`, indicating that the selected mark start location is 5.8% of the waveform.

MARK:SELECTED:STATE

This command sets or queries the on or off state of the selected mark. The selected mark is at or near the center of the display. If you push the Set/Clear button, this mark will disappear.

Group Search and Mark

Syntax

```
MARK:SELECTED:STATE {<NR1>|OFF|ON}
MARK:SELECTED:STATE?
```

Arguments <NR1> = 0 disables the selected mark; any other value enables the selected mark. For queries, a 0 is returned if the selected mark state is off; a 1 is returned if the selected mark state is on.

OFF argument disables selected mark.

ON argument enables the selected mark.

Examples MARK:SELECTED:STATE ON turns the selected mark state on.

MARK:SELECTED:STATE? might return :MARK:SELECTED:STATE 0, indicating the selected mark stare is off.

MARK:SELECTED:ZOOm:POSition? (Query Only)

This query-only command returns the position of the selected mark, of the zoom overview window. If the return value is less than zero then the zoom scale will not be changed when Next or Prev makes this the selected mark.

Group Search and Mark

Syntax

```
MARK:SELECTED:ZOOm:POSition?
```

Examples MARK:SELECTED:ZOOM:POSITION? might return :MARK:SELECTED:ZOOM:POSITION 27.7000, indicating that the position is 27.7%.

MARK:SELECTED:ZOOm:SCAle? (Query Only)

This query-only command returns the scale of the selected mark of the zoom overview window. The returned value might be < 0 (for example from Search),

which means that the zoom scale will not be changed when Next or Prev makes this the selected mark.

Group Search and Mark

Syntax `MARK:SELECTED:ZOOm:SCAle?`

Examples `MARK:SELECTED:ZOOM:SCALE?` might return `:MARK:SELECTED:ZOOM:SCAle 5.0`, indicating that the zoom scale value 5.0 will be used when a mark is the selected mark.

MARK:TOTal? (Query Only)

This query-only command returns how many marks are currently in use.

Group Search and Mark

Syntax `MARK:TOTal?`

Examples `MARK:TOTAL?` might return `:MARK:TOTAL 210`, indicating that 210 marks are in use.

MASK? (Query Only)

This query-only command returns the states of all settable mask parameters.

Group Mask

Syntax `MASK?`

Examples `MASK?` might return the following response. `:MASK:FILTER 0;AUTOADJUST 0;AUTOADJUST:HDELTA 10;VDELTA 10;:MASK:AUTOSET:AUTOADJUST 1;HPOS 1;HSCALE 1;MODE MANUAL;OFFSETADJ 1;TRIGGER 1;VPOS 1;VSCALE 1;STANDARD NONE;USER:ONE 0.0000;ZERO 0.0000;TYPE NORMALIZED;:MASK:SOURCE CH1;COUNT:FAILURES 0;HITS 0;SEG1:.`

MASK:AUTOAdjust

This command optimizes or queries the signal position within the mask to minimize hits. It sets a mode so that the MASK:SOURCE waveform is compared against the mask and is shifted up, down, left, or right to minimize the hits.

Group Mask

Syntax

```
MASK:AUTOAdjust {<NR1>|OFF|ON}
MASK:AUTOAdjust?
```

Arguments

<NR1> = 0 disables the autoadjust function; any other value enables it.

ON enables the autoadjust function.

OFF disables the autoadjust function.

Examples

MASK:AUTOADJUST ON enables the autoadjust function.

MASK:AUTOADJUST? might return :MASK:AUTOADJUST 1, indicating that the autoadjust function is enabled.

MASK:AUTOAdjust:HDELTA

This command sets or returns how far autofit searches horizontally.

Autofit moves the waveform vertically and/or horizontally to reduce the number of hits within a given mask. If infinite or variable persistence is enabled, these movements will clear any persistent data. If Autofit makes frequent adjustments, there might be very little or even no persistent data displayed.

Group Mask

Syntax

```
MASK:AUTOAdjust:HDELTA {<NR3>}
MASK:AUTOAdjust:HDELTA?
```

Arguments

<NR3> is a floating point number that represents a percent of a division.

Examples

MASK:AUTOADJUST:HDELTA 0.6 sets the horizontal autofit search to 0.6 division.

MASK:AUTOADJUST:HDELTA? might return :MASK:AUTOADJUST:HDELTA 2, indicating that the horizontal autofit search is set to 2 divisions.

MASK:AUTOAdjust:VDELTA

This command sets or returns how far autofit searches vertically.

Autofit moves the waveform vertically and/or horizontally to reduce the number of hits within a given mask. If infinite or variable persistence is enabled, these movements will clear any persistent data. If Autofit makes frequent adjustments, there might be very little or even no persistent data displayed.

Group Mask

Syntax

```
MASK:AUTOAdjust:VDELTA {<NR3>}
MASK:AUTOAdjust:VDELTA?
```

Arguments <NR3> is a floating point number that represents a percent of a division.

Examples MASK:AUTOADJUST:VDELTA 0.6 sets the vertical autofit search to 0.6 of a division.

MASK:AUTOADJUST:VDELTA? might return :MASK:AUTOADJUST:VDELTA 2, indicating that the vertical autofit search is set to 2 divisions.

MASK:AUTOSet:AUTOAdjust

NOTE. *The command works only if the Option MTM is enabled.*

This command sets or returns whether the instrument settings are automatically matched to signal characteristics and specific mask requirements. It controls what happens at the end of a mask autoset. If ON, the horz/vert auto adjustment is run ONCE to potentially improve the result of mask autoset.

This command, like all the MASK:AUTOSET commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group Mask

Syntax

```
MASK:AUTOSet:AUTOAdjust {<NR1>|OFF|ON}
MASK:AUTOSet:AUTOAdjust?
```

Arguments

<NR1> = 0 disables the autoset:autoadjust function; any other value enables it.

ON enables the autoset:autoadjust function.

OFF disables the autoset:autoadjust function.

Examples

MASK:AUTOSET:AUTOADJUST ON enables the autoset autoadjust function.

MASK:AUTOSET:AUTOADJUST? might return :MASK:AUTOSET:AUTOADJUST 1, indicating that the autoset:autoadjust function is enabled.

MASK:AUTOSet:HPOS

This command controls whether the autoset algorithm will attempt to change the horizontal position of the signal while attempting to do a mask autoset. The default is ON.

This command, like all the MASK:AUTOSET commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group

Mask

Syntax

```
MASK:AUTOSet:HPOS {<NR1>|OFF|ON}
MASK:AUTOSet:HPOS?
```

Arguments

<NR1> = 0 disables the autoset:hpos function; any other value disables enables it.

ON enables the autoset:hpos function.

OFF disables the autoset:hpos function.

Examples

MASK:AUTOSET:HPOS ON enables the autoset:hpos function.

MASK:AUTOSET:HPOS? might return :MASK:AUTOSET:HPOS 1, indicating that the autoset:hpos function is enabled.

MASK:AUTOSet:HSCAle

This command controls whether the autoset algorithm will attempt to change the horizontal scale while attempting a mask autoset.

This command, like all the MASK:AUTOSET commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group

Mask

Syntax

```
MASK:AUTOSet:HSCAle {<NR1>|OFF|ON}
MASK:AUTOSet:HSCAle?
```

Arguments

<NR1> = 0 disables the autoset:hscale function; any other value enables it.

ON enables the autoset:hscale function.

OFF disables the autoset:hscale function.

Examples

MASK:AUTOSET:HSCALE ON enables the autoset:hscale function.

MASK:AUTOSET:HSCALE? might return :MASK:AUTOSET:HSCALE 1, indicating that the autoset:hscale function is enabled.

MASK:AUTOSet:MODe

This command controls whether a mask autoset will be automatically done after a standard mask is selected. The autoset will never happen if the standard mask is selected from GPIB, since a "SET?" sent back to the oscilloscope should not perform a mask autoset.

The default value is: manual.

This command, like all the MASK:AUTOSET commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group

Mask

Syntax

```
MASK:AUTOSet:MODe {MANual|AUTO}
MASK:AUTOSet:MODe?
```

Arguments

MANual mask autoset is set manually.

AUTO mask autoset is set automatically.

Examples

MASK:AUTOSET:MODE AUTO sets the mask autoset manually.

MASK:AUTOSET:MODE? might return :MASK:AUTOSET:MODE MANUAL, indicating that the mask autoset is set manually.

MASK:AUTOSet:OFFSETAdj

This command sets mask autoset not to enforce the rule that, for pulse standards, require 0 V to be in a certain place in the mask. Instead, mask autoset will attempt

to measure the DC offset in the signal and use oscilloscope offset or position controls to compensate for the DC offset in the signal.

This command, like all the MASK:AUTOSET commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group Mask

Syntax

```
MASK:AUTOSet:OFFSETAdj {<NR1>|OFF|ON}
MASK:AUTOSet:OFFSETAdj?
```

Arguments

<NR1> = 0 disables the autoset:offsetadj function; any other value enables it.

ON enables the autoset:offsetadj function.

OFF disables the autoset:offsetadj function.

Examples

MASK:AUTOSET:OFFSETADJ ON enables the autoset:offsetadj function.

MASK:AUTOSET:OFFSETADJ? might return :MASK:AUTOSET:OFFSETADJ 1, indicating that the autoset:offsetadj function is enabled.

MASK:AUTOSet:STANdard

NOTE. *The command works only if the Option MTM is enabled.*

This command selects the standard mask in a mask autoset.

This command, like all the MASK:AUTOSET commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group Mask

Syntax

```
MASK:AUTOSet:STANdard
{ATATXG1|ATATXG2|ATATXG3|ATARXG1|ATARXG2
|ATARXG3|CLOCKCoax|CLOCKSymmetrical|D1|D2
|DS0Contra|DS0Double|DS0Single|DS0Timing|DS1
|DS1A|DS1C|DS2RATESymmetrical|DS2RATECoax
|DS2|DS3|DS4NA|DS4NA_Max
|E1Symmetrical|E1Coax|E2|E3|E4_0|E4_1|ENET100FX
|ENET100STP |ENET100UTP |ENET1250|ENET1000BCX_NTP2
|ENET1000BCX_ATP2|ENET1000BCX_ATP3|ENETXAUI_Near
|ENETXAUI_Far|FC133|FC266|FC531|FC1063|FC1063Draft
```

```
|FC2125|FC133E|FC266E|FC531E|FC1063E|FC1063E_NBT
|FC1063E_NDT|FC1063E_NGT|FC1063E_ABT|FC1063E_ADT
|FC1063E_AGT|FC1063E_ABR|FC1063E_ADR|FC1063E_AGR
|FC2125E_NBT|FC2125E_NDT|FC2125E_NGT|FC2125E_ABT
|FC2125E_ADT|FC2125E_AGT|FC2125E_ABR|FC2125E_ADR
|FC2125E_AGR|FC4250E_ABR|FC4250E_ABT|FC4250E_ADR
|FC4250E_ADT|FC4250E_AGR|FC4250E_AGT|FC4250E_NBT
|FC4250E_NDT|FC4250E_NGT|FST1|FST2|FST3|FST4
|FST5|FST6|FW1394BS400BT1|FW1394BS400BT2
|FW1394BS800BT1|FW1394BS800BT2|FW1394BS1600BT1
|FW1394BS1600BT2|FW1394BS400B|FW1394BS800B
|FW1394BS1600B|G703DS1|G703DS3|HST1|HST2
|HST3|HST4|HST5|HST6|INF2_5G
|INF2_5GE|NONe|OC1|OC3|OC12|OC48|OC48_FEC
|RATE32Mbit|RATE97Mbit|RIO_DRV500M|RIO_DRV750M
|RIO_DRV1G|RIO_DRV1_5G|RIO_DRV2G|RIO_EDRV500M
|RIO_EDRV750M |RIO_EDRV1G|RIO_EDRV1_5G|RIO_EDRV2G
|RIO_RCV500M|RIO_RCV750M|RIO_RCV1G|RIO_RCV1_5G
|RIO_RCV2G|RIO_SERIAL_1G|RIO_SERIAL_2G|RIO_SERIAL_3G
|SFI5_XMITADATA2|SFI5_XMITCDATA2|SFI5_XMITACLK2
|SFI5_XMITCCLK2|SFI5_RCVBDATA2|SFI5_RCVDDATA2
|SFI5_RCVBCLK2|SFI5_RCVDCLK2|SFI5_XMITADATA3
|SFI5_XMITCDATA3|SFI5_XMITACLK3|SFI5_XMITCCLK3
|SFI5_RCVBDATA3|SFI5_RCVDDATA3|SFI5_RCVBCLK3
|SFI5_RCVDCLK3|PCIEXPRESS_Xmit|PCIEXPRESS_Rcv
|SAS1_5_IR|SAS1_5_CR|SAS1_5_XR|SAS1_5_IR_AASJ
|SAS1_5_CR_AASJ|SAS1_5_XR_AASJ|SAS1_5_SATA
|SAS3_0_IR|SAS3_0_CR|SAS3_0_XR|SAS3_0_IR_AASJ
|SAS3_0_CR_AASJ|SAS3_0_XR_AASJ|SAS3_0_SATA
|STM0_1|STM0_0|STM0_HDBX|STM1E_0|STM1E_1|STS1Pulse
|STS1Eye|STS3|STS3_Max|TFI5_2|TFI5_3
|USERMask|VIDEO270|VIDEO292M|VIDEO360|VSROC192}
MASK:AUTOSet:STANdard?
```

Arguments

`ATATXG1` (Serial ATA, G1 Tx, 1.5 Gb/s)

`ATATXG2` (Serial ATA, G2 Tx, 3.0 Gb/s)

`ATATXG3` (Serial ATA, G3 Tx).

`ATARXG1` (Serial ATA, G1 Rx 1.5 Gb/s)

`ATARXG2` (Serial ATA, G2, Rx, 3.0 Gb/s)

`ATARXG3` (Serial ATA, G3, Rx)

`G703D1` (ITU-T, G703 (10/98), DS1 Rate, 1.544 Mb/s)

`DS1` (ANSI T1.102-1993 (R1999), DS1, 1.544 Mb/s)

`DS1A` (ANSI T1.102-1993 (R1999), DS1A, 2.048 Mb/s)

`DS1C` (ANSI T1.102-1993 (R1999), DS1C, 3.152 Mb/s)

DS2 (ANSI T1.102-1993 (R1999), DS2, 6.312 Mb/s)

DS3 (ANSI T1.102-1993 (R1999), DS3, 44.736 Mb/s)

DS4NA (ANSI T1.102-1993 (R1999), DS4NA, 139.26 Mb/s)

DS4NA_Max (ANSI T1.102-1993 (R1999), DSNA Max Output, 139.26 Mb/s)

DS2RATECoax (ITU-T, G703 (10/98), D2 Rate Coax, 6.312 Mb/s)

DS2RATESymmetrical (ITU-T, G703 (10/98), D2 Rate Sym, 6.312 Mb/s)

E1Coax (ITU-T, G703 (10/98), E1 Coax Pair, 2.048 Mb/s)

E1Symmetrical (ITU-T, G703 (10/98), E1 Sym Pair, 2.048 Mb/s)

E2 (ITU-T, G703 (10/98), 8.448 Mb/s)

RATE32Mbit (ITU-T, G703 (10/98), 32.064 Mb/s)

E3 (ITU-T, G703 (10/98), E3, 34.368 Mb/s)

E4_0 (ITU-T, G703 (10/98), E4 Binary 0)

E4_1 (ITU-T, G703 (10/98), E4 Binary 1)

ENET100STP (IEEE Std 802.3 and ANSI X3.263-1995, 100 Base-Tx, STP, 125 Mb/s)

ENET100UTP (IEEE Std 802.3 and ANSI X3.263-1995, 100 Base-Tx, UTP, 125 Mb/s)

ENET1250 (IEEE Std 802.3 and ANSI X3.263-1995, GB Ethernet, 1.25 Gb/s)

FC133 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 132.8 Mb/s)

FC266 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 265.6 Mb/s)

FC531 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 531.2 Mb/s)

FC1063 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 1.065 Gb/s)

FC1063Draft (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, Draft Rev 11)

FC2125 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 2.125)

FC133E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical 132.8 Mb/s)

FC266E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical, 132.8 Mb/s)

FC531E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical, 531.2 Mb/s)

FC1063E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical, 1.0625 Gb/s)

FC1063E_NBT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Beta, Transm)

FC1063E_NDT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Delta, Transm)

FC1063E_NGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Gamma, Transm)

FC1063E_ABT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Transm)

FC1063E_ADT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Transm)

FC1063E_AGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Transm)

FC1063E_ABR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Recv)

FC1063E_ADR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Delta, Recv)

FC1063E_AGR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Recv)

FC2125E_NBT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Beta, Transm)

FC2125E_NDT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Delta, Transm)

FC2125E_NGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Gamma, Transm)

FC2125E_ABT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Transm)

FC2125E_ADT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Delta, Transm)

FC2125E_AGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Transm)

FC2125E_ABR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Recv)

FC2125E_ADR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Delta, Recv)

FC2125E_AGR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Recv)

ENET1000BCX_NTP2 (1000B-CX Norm, TP2, 1.25 Gb/s)

ENET1000BCX_ATP2 (1000B-CX Abs, TP2, 1.25 Gb/s)

ENET1000BCX_ATP3 (1000B-CX Abs, TP3, 1.25 Gb/s)

ENETXAUI_Near (10 Gigabit Attachment Unit Interface (XAUI), Near, 3,125 Gb/s)

ENETXAUI_Far (10 Gigabit Attachment Unit Interface (XAUI), Far, 3.125 Gb/s)

FST1 (USB, FS:T1, 12 Mb/s)

FST2 (USB, FS:T2, 12 Mb/s)

FST3 (USB, FS:T3, 12 Mb/s)

FST4 (USB, FS:T4: 12 Mb/s)

FST5 (USB, FS:T5, 12 Mb/s)

FST6 (USB, FS:T6, 12 Mb/s)

FW1394BS400BT1 (IEEE 1394b, S400b T1, 491.5 Mb/s)

FW1394BS400BT2 (IEEE 1394b, S400b T2, 491.5 Mb/s)

FW1394BS800BT1 (IEEE 1394b, S800b T1, 983.0 Mb/s)

FW1394BS800BT2 (IEEE 1394b, S800b T2, 983.0 Mb/s)

FW1394BS1600BT1 (IEEE 1394b, S1600b T1, 1.966 Gb/s)

FW1394BS1600BT2 (IEEE 1394b, S1600b T2, 1.966 Gb/s)

FW1394BS400B (IEEE 1394b, S400 Optical, 491.5 Mb/s)

FW1394BS800B (IEEE 1394b, S800 Optical, 988.0 Mb/s)

FW1394BS1600B (IEEE 1394b, S1600 Optical, 1.966 Gb/s)

G703DS3 (ITU-T, G703 (10/98))

HST1 (USB, HS:T1, 480 Mb/s)

HST2 (USB, HS:T2, 480 Mb/s)

HST3 (USB, HS:T3, 480 Mb/s)

HST4 (USB, HS:T4, 480 Mb/s)

HST5 (USB, HS:T5, 480 Mb/s)

HST6 (USB, HS:T6, 480 Mb/s)

INF2_5G (InfiniBand, IBTA Spec 1.0a, 2.5 Optical, 2.5 Gb/s)

INF2_5GE (InfiniBand, IBTA Spec 1.0a, 2.5 Electrical, 2.5 Gb/s)

OC1 (GR 253-CORE Issue 3 9/21/2000 OC1/STM0, 51.84 Mb/s)

OC3 (GR 253-CORE Issue 3 9/21/2000 OC1/STM1, 155.52, Mb/s)

OC12 (GR 253-CORE Issue 3 9/21/2000 OC1/STM4, 622.08 Mb/s)

OC48 (GR 253-CORE Issue 3 9/21/2000 OC1/STM16, 2.4883 Gb/s

OC48_FEC (Forward Error Correction - CSA8000 mask, 2.666 Gb/s)

PCIEXPRESS_Xmit (PCI Express Transmitter, 2.5 Gb/s)

PCIEXPRESS_Rcv (PCI Express Receiver, 2.5 Gb/s)

RATE97Mbit (ITU-T, G703 (10/98), 97 Mbit, 97.728 Mb/s)

RIO_DRV500M (Rapid IO Driver, 500 Mb/s)

RIO_DRV750M (Rapid IO Driver, 750 Mb/s)

RIO_DRV1G (Rapid IO Driver, 1 Gb/s)

RIO_DRV1_5G (Rapid IO Driver, 5 Gb/s)

RIO_DRV2G (Rapid IO Driver, 2 Gb/s)

RIO_EDRV500M (Rapid IO Extended Driver, 500 Mb/s)

RIO_EDRV750M (Rapid IO Extended Driver, 750 Mb/s)

RIO_EDRV1G (Rapid IO Extended Driver, 1 Gb/s)

RIO_EDRV1_5G (Rapid IO Extended Driver, 1.5 Gb/s)

RIO_EDRV2G (Rapid IO Extended Driver, 2 Gb/s)

RIO_RCV500M (Rapid IO Receiver, 500 Mb/s)

RIO_RCV750M (Rapid IO Receiver, 750 Mb/s)

RIO_RCV1G (Rapid IO Receiver, 1 Gb/s)

RIO_RCV1_5G (Rapid IO Receiver, 1.5 Gb/s)

RIO_RCV2G (Rapid IO Receiver, 2 Gb/s)

RIO_SERIAL_1G (Rapid IO Serial, 1.25 Gb/s)

RIO_SERIAL_2G (Rapid IO Serial, 2.5 Gb/s)

RIO_SERIAL_3G (Rapid IO Serial, 3.25 Gb/s)

SFI5_XMITADATA2 (SFI15 Transmit: Test Point A Data Signal 2, 2.488 Gb/s)

SFI5_XMITCDATA2 (SFI15 Transmit: Test Point C Data Signal 2, 2.488 Gb/s)

SFI5_XMITACLK2 (SFI15 Transmit: Test Point A Clock Signal 2, 2.488 Gb/s)

SFI5_XMITCCLK2 (SFI15 Transmit: Test Point C Clock Signal 2, 2.488 Gb/s)

SFI5_RCVBDATA2 (SFI15 Receive: Test Point B Data Signal 2, 2.488 Gb/s)

SFI5_RCVDDATA2 (SFI15 Receive: Test Point D Data Signal 2, 2.488 Gb/s)

SFI5_RCVBCLK2 (SFI15 Receive: Test Point B Clock Signal 2, 2.488 Gb/s)

SFI5_RCVDCLK2 (SFI15 Receive: Test Point D Clock Signal 2, 2.488 Gb/s)

SFI5_XMITADATA3 (SFI15 Transmit: Test Point A Data Signal 3, 3.125 Gb/s)

SFI5_XMITCDATA3 (SFI15 Transmit: Test Point C Data Signal 3, 3.125 Gb/s)

SFI5_XMITACLK3 (SFI15 Transmit: Test Point A Clock Signal 3, 3.125 Gb/s)

SFI5_XMITCCLK3 (SFI15 Transmit: Test Point C Clock Signal 3, 3.125 Gb/s)

SFI5_RCVBDATA3 (SFI15 Receive: Test Point B Data Signal 3, 3.125 Gb/s)

SFI5_RCVDDATA3 (SFI15 Receive: Test Point D Data Signal 3, 3.125 Gb/s)

SFI5_RCVBCLK3 (SFI15 Receive: Test Point B Clock Signal 3, 3.125 Gb/s)

SFI5_RCVDCLK3 (SFI15 Receive: Test Point D Clock Signal 3, 3.125 Gb/s

STM1E_0 (ITU-T, G703 (10/98), STM1E Binary 0)

STM1E_1 (ITU-T, G703 (10/98), STM1E Binary 1)

STS1Pulse (ANSI T1.102-1993 (R1999), STS-1 Pulse, 51.84 Mb/s)

STS1Eye (ANSI T1.102-1993 (R1999), STS-1 Eye, 51.84 Mb/s)

STS3 (ANSI T1.102-1993 (R1999), STS-3, 155.52 Mb/s)

STS3_Max (ANSI T1.102-1993 (R1999), STS-3 Max Output, 155.52 Mb/s)

TFI15_2 (TFI-5, 2.488 Gb/s)

TFI5_3 (TFI-5, 3.1104 Gb/s)

USERMask

VIDEO292M (SMPTE, 1.485 Gb/s)

VSROC192 (VSR OC192/STM64, 1.24416 Gb/s)

Examples

MASK:AUTOSET:STANDARD E1COAX selects the E1 Coax standard mask.

MASK:AUTOSET:STANDARD? might return :MASK:AUTOSET:STANDARD E1COAX, indicating that the E1 Coax mask is selected as the current mask standard.

MASK:AUTOSet:TRIGger

This command specifies whether a mask AUTOSet affects the trigger level. Other trigger settings such as type of trigger are not changed by autoset. The default is ON.

This command, like all the mask autoset commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group

Mask

Syntax

```
MASK:AUTOSet:TRIGger {<NR1>|OFF|ON}
MASK:AUTOSet:TRIGger?
```

Arguments

<NR1> = 0 disables the autoset:trigger function; any other value enables it.

ON enables the autoset:trigger function.

OFF disables the autoset:trigger function.

Examples

MASK:AUTOSET:TRIGGER ON enables the autoset:trigger function.

MASK:AUTOSET:TRIGGER? might return :MASK:AUTOSET:TRIGGER 1, indicating that the autoset:trigger function is enabled.

MASK:AUTOSet:USER:ONE

This command sets or queries the level 1 value used by autoset for user masks.

Group Mask

Syntax
```
MASK:AUTOSet:USER:ONE <NR3>
MASK:AUTOSet:USER:ONE?
```

Related Commands MASK:AUTOSet:USER:TYPe, MASK:AUTOSet:USER:ZERo

Arguments <NR3> sets the level 1 value used by autoset for user masks.

Examples MASK:AUTOSET:USER:ONE 1 sets the autoset for 1 in user masks is 1.0 volts.

MASK:AUTOSET:USER:ONE? might return :MASK:AUTOSET:USER:ON 0.0000, indicating that the autoset is using 0.0 volts for 1 in user masks.

MASK:AUTOSet:USER:TYPe

This command sets or queries how autoset scales a mask.

Group Mask

Syntax
```
MASK:AUTOSet:USER:TYPe {ABSolute|NORMALIZed}
MASK:AUTOSet:USER:TYPe?
```

Related Commands MASK:AUTOSet:USER:ONE, MASK:AUTOSet:USER:ZERo

Arguments ABSOLUTE tells autoset to use the values supplied by MASK:AUTOSet:USER:ONE and MASK:AUTOSet:USER:ZERo for the one and zero of the mask.

NORMALIZED tells autoset to determine the mask one and zero by looking at the mask.

Examples MASK:AUTOSET:USER:TYPE NORMALIZED sets the autoset scales in a mask to normalize.

MASK:AUTOSET:USER:TYPE? might return :MASK:AUTOSET:USER:TYPE NORMALIZED, indicating that the autoset is set to normalize scales in a mask.

MASK:AUTOSet:USER:ZERo

NOTE. *The command works only if the Option MTM is enabled.*

This command sets or queries the level 0 used by autoset for user masks.

Group Mask

Syntax
```
MASK:AUTOSet:USER:ZERo <NR3>
MASK:AUTOSet:USER:ZERo?
```

Related Commands MASK:AUTOSet:USER:TYPe, MASK:AUTOSet:USER:ONE

Arguments <NR3> sets the zero level that autoset uses for user masks.

Examples MASK:AUTOSET:USER:ZERO 1.000 sets the autoset for zero in user masks to scales in a mask to normalize.

MASK:AUTOSET:USER:ZERO? might return :MASK:AUTOSET:USER:ZERO 0.0000, indicating that the autoset is using 0.0 volts for zero in user masks.

MASK:AUTOSet:VPOS

This command controls whether the mask autoset algorithm will affect the vertical position (or offset) of the signal while attempting to autoset. The default is ON.

This command, like all the mask autoset commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group Mask

Syntax
```
MASK:AUTOSet:VPOS {<NR1>|OFF|ON}
MASK:AUTOSet:VPOS?
```

Arguments <NR1> = 0 disables the autoset:vpos function; any other value enables it.

ON enables the autoset:vpos function.

OFF disables the autoset:vpos function.

Examples MASK:AUTOSET:VPOS ON enables the autoset:vpos function.

MASK:AUTOSET:VPOS? might return :MASK:AUTOSET:VPOS 0.0000, indicating that the autoset:vpos function is disabled.

MASK:AUTOSet:VSCAle

NOTE. *The command works only if the Option MTM is enabled.*

This command controls whether the mask autoset algorithm will affect the vertical scale while attempting to autoset.

This command, like all the mask autoset commands, affects only an autoset on a standard mask, not the general instrument autoset function.

Group Mask

Syntax
```
MASK:AUTOSet:VSCAle {<NR1>|OFF|ON}
MASK:AUTOSet:VSCAle?
```

Arguments <NR1> = 0 disables the autoset:vscale function; any other value enables it.

ON enables the autoset:vscale function.

OFF disables the autoset:vscale function.

Examples MASK:AUTOSET:VSCALE ON enables the autoset:vscale function.

MASK:AUTOSET:VSCALE? might return :MASK:AUTOSET:VSCALE 0.0000, indicating that the autoset:vcale function is disabled.

MASK:COPy:USER (No Query Form)

This command copies the currently displayed mask to the user mask.

Group Mask

Syntax MASK:COPy:USER

Examples MASK:COPY:USER copies the current mask to the user mask.

MASK:COUNt (No Query Form)

NOTE. *The command works only if the Option MTM is enabled.*

This command resets to zero the number of hits for all mask segments on the specified source.

This is equivalent to pressing the front panel 'Clear' button.

Group Mask

Syntax `MASK:COUNt RESET`

Arguments `RESET` sets to zero the number of hits for all mask segments on the specified source.

Examples `MASK:COUNT` resets to zero the number of hits for all mask segments on the specified source.

MASK:COUNt:FAILURES? (Query Only)

This query-only command returns the number of pass/fail tests that have failed. The default is 0. Pass/fail testing must be enabled.

Group Mask

Syntax `MASK:COUNt:FAILURES?`

Returns `<NR1>`

Examples `MASK:COUNT:FAILURES?` might return `:MASK:COUNT:FAILURES 0`, showing that no tests have failed.

MASK:COUNt:HITS? (Query Only)

NOTE. *The command works only if the Option MTM is enabled.*

This query-only command returns the number of hits on all mask segments. Hit counting must be turned on.

Group Mask

Syntax `MASK:COUNt:HITS?`

Returns `<NR1>`

Examples `MASK:COUNT:HITS?` might return `:MASK:COUNT:HITS 0`, showing there were no hits from any source on any mask segments.

MASK:COUNt:SEG<m>:HITS? (Query Only)

This query-only command returns the number of hits in mask segment <m>. Hit counting must be turned on.

m is the mask segment number.

Group Mask

Syntax `MASK:COUNt:SEG<m>:HITS?`

Returns `<NR1>` returns the number of hits in the mask segment.

Examples `MASK:COUNT:SEG1:HITS?` might return `:MASK:COUNT:SEG1:HITS 0`, showing that there have been no hits.

MASK:COUNt:STATE

This command sets or queries the mask hits count state; it controls whether mask counting is being done. MASK:DISplay must be ON to enable `MASK:COUNt:STATE` to count mask violations.

Group Mask

Syntax
```
MASK:COUNt:STATE {<NR1>|OFF|ON}
MASK:COUNt:STATE?
```

Arguments S123hi456va

<NR1> = 0 turns off mask hit counting, and other values turn on mask hit counting.

ON turns on mask counting.

OFF turns off mask counting. This is the default state.

Examples MASK:COUNT:STATE ON turns on mask hit counting.

MASK:COUNT:STATE? might return :MASK:COUNT:STATE 1, indicating that the mask counting is on.

MASK:COUNt:TESTS? (Query Only)

This query-only command returns the number of pass/fail tests that have occurred.

Group Mask

Syntax MASK:COUNt:TESTS?

Returns <NR1>

Examples MASK:COUNT:TESTS? might return :MASK:COUNT:TESTS 50, showing 50 tests have occurred.

MASK:COUNt:TOTal? (Query Only)

This query-only command returns the sum of all hits in all mask segments. This command is the same as MASK:COUNT:HITS? and is kept for compatibility with other Tektronix instruments.

Group Mask

Syntax MASK:COUNt:TOTal?

Returns <NR1>

Examples MASK:COUNT:TOTAL? might return :MASK:COUNT:TOTAL 50, showing 50 hits have occurred.

MASK:COUNt:VIOLATIONS? (Query Only)

This query-only command returns the number of test violations that have occurred in the current pass/fail test. A test violation occurs when any part of a waveform falls within any mask segment. The default is 0.

Group Mask

Syntax `MASK:COUNt:VIOLATIONS?`

Related Commands MASK:COUNt:TESTS?, MASK:COUNt:TOTal?, MASK:COUNt:WAVEFORMS?, MASK:TESt:REPeat

Returns `<NR1>`

Examples `MASK:COUNT:VIOLATIONS?` might return `:MASK:COUNT:VIOLATIONS 50`, showing that 50 test violations have occurred.

MASK:COUNt:WAVEFORMS? (Query Only)

This query-only command returns the number of waveforms that have been acquired and processed during pass/fail testing.

Group Mask

Syntax `MASK:COUNt:WAVEFORMS?`

Related Commands MASK:COUNt:TESTS?, MASK:COUNt:TOTal?, MASK:COUNt:VIOLATIONS?

Returns `<NR1>`

Examples `MASK:COUNT:WAVEFORMS?` might return `:MASK:COUNT:WAVEFORMS 1000`, showing that 1,000 waveforms have been acquired and processed.

MASK:DISplay

This command sets or queries whether defined masks are displayed on the screen. This is useful for temporarily turning off user-defined masks without deleting them. It is also useful for removing a standard mask from the screen, but leaving it as the selected standard. Mask counting, mask testing, and mask autoset are unavailable if the mask display is Off. The default is On.

Group Mask

Syntax

```
MASK:DISplay {<NR1>|OFF|ON}
MASK:DISplay?
```

Arguments

<NR1> = 0 removes the masks from the display; any other value shows the masks on the display.

ON shows the masks on the display. This is the default value.

OFF removes the masks from the display.

Examples

MASK:DISPLAY ON sets the display to show the defined masks.

MASK:DISPLAY? might return :MASK:DISPLAY 1, indicating that the defined masks on the screen are displayed.

MASK:FILTer

This command sets or returns whether the selected digital filter will be run on the waveform data. The filter simulates optical hardware. That is, it simulates different hardware for each of several different optical standards.

The digital filter runs on OC1, OC3, OC12, OC48, FC133, FC266, FC531, FC1063, FC2125Draft, Gigabit Ethernet, Infiniband 2.5 Gb, 1394 b, 393 Mb, 786.43 Mb, 1.572 Gb

Group Mask

Syntax

```
MASK:FILTer {<NR1>|OFF|ON}
MASK:FILTer?
```

Related Commands ACQuire:MODe

Arguments

<NR1> = 0 disables the digital filter; any other value enables it.

OFF disables the digital filter.

ON enables the digital filter.

Examples

MASK:FILTERON enables the digital filter.

MASK:FILTER? might return :MASK:FILTER 1, indicating that the selected digital filter will be run on the waveform data.

MASK:HIGHLIGHTHits

This command sets or returns whether hits in a mask are highlighted in different colors than other waveform data. The default is On.

Group

Mask

Syntax

```
MASK:HIGHLIGHTHits {<NR1>|OFF|ON}
MASK:HIGHLIGHTHits?
```

Arguments

<NR1> = 0 disables the mask:highlighthits function; any other value enables it.

OFF disables the mask:highlighthits function.

ON enables the mask:highlighthits function.

Examples

MASK:HIGHLIGHTHITS ON enables the mask highlighthits function.

MASK:HIGHLIGHTHITS? might return :MASK:HIGHLIGHTHITS 1, indicating that the mask:highlighthits function is enabled.

MASK:INVert

This command controls whether the mask is drawn inverted. It has no effect if this mask cannot be inverted. The default is Off (Positive).

Group

Mask

Syntax

```
MASK:INVert {<NR1>|OFF|ON}
```

Arguments <NR1> = 0 disables the mask:invert function; any other value enables it.

OFF Positive

ON Negative

Examples MASK:INVERTON enables the mask invert function.

MASK:INVERT? might return :MASK:INVERT 1, indicating that the mask is drawn inverted.

MASK:LOCk

This command sets or returns the mask lock to waveform mode. It locks the mask to the waveform such that changes to the horizontal and/or vertical scale settings redraws the mask segments in proportion. This feature allows for expanding horizontal and/or vertical settings to zoom in on waveforms and masks to visually examine violation areas in more detail.

Group Mask

Syntax
```
MASK:LOCk {<NR1>|OFF|ON}
MASK:LOCk?
```

Arguments <NR1> = 0 turns off mask lock to waveform and any other value turns on mask lock to waveform.

OFF turns off mask lock to waveform.

ON turns on mask lock to waveform.

Examples MASK:LOCK ON turns on mask lock to waveform.

MASK:LOCK? might return :MASK:LOCK 1, indicating that the mask lock to waveform mode is enabled.

MASK:MARgin:PERCent

This command sets or queries the mask margin percentage. If you set a positive mask margin percentage, the polygon expands and the mask test will be more difficult to pass due to the expanded test area. If you set a negative mask margin percentage, the polygon shrinks and the mask test will be easier to pass due to the smaller area.

Group Mask

Syntax
```
MASK:MARgin:PERCent <NR3>
MASK:MARgin:PERCent?
```

Arguments <NR3> values range from -50.0 to +50.0. The default is 5.

Examples MASK:MARGIN:PERCENT 25 sets the mask margin percentage to 25.

MASK:MARGIN:PERCENT? might return :MASK:MARGIN:PERCENT 1.00000000000E+001, indicating that the mask margin is set to 10%.

MASK:MARgin:STATE

This command sets or returns the state of the mask margins.

Group Mask

Syntax
```
MASK:MARgin:STATE {<NR1>|OFF|ON}
MASK:MARgin:STATE?
```

Arguments <NR1> = 0 turns off mask margins; any other integer turns on the selected mask margins.

OFF turns off mask margins. The currently displayed margined mask is erased and the original mask is displayed.

ON turns on mask margins.

Examples MASK:MARGIN:STATE ON turns on mask margins.

MASK:MARGIN:STATE? might return :MASK:MARGIN:STATE 1, indicating that the mask margins are on.

MASK:MASKPRE:AMPlitude

This command sets or returns the current mask's nominal pulse amplitude in volts. The query form of this command returns the nominal pulse amplitude of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax
```
MASK:MASKPRE:AMPlitude <NR3>
MASK:MASKPRE:AMPlitude?
```

Related Commands MASK:MASKPRE:VSCAle

Arguments <NR3> is a floating number that sets the nominal pulse amplitude in volts.

Examples MASK:MASKPRE:AMPLITUDE 2.4 sets the current mask nominal pulse amplitude to 2.4 volts.

MASK:MASKPRE:AMPLITUDE? might return :MASK:MASKPRE:AMPLITUDE 1, indicating that the current mask's nominal pulse amplitude is set to 1.0 V.

MASK:MASKPRE:HSCAle

This command sets or returns the nominal timing resolution used to draw the mask in time/division. The query form of this command returns the nominal timing resolution of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax
```
MASK:MASKPRE:HSCAle <NR3>
MASK:MASKPRE:HSCAle?
```

Related Commands MASK:MASKPRE:HTRIGPOS

Arguments <NR3> is a floating point number that sets the mask drawing timing resolution.

Examples MASK:MASKPRE:HSCALE 40E-9 sets the nominal timing resolution used to draw the mask pulse shape to 40 nanoseconds per division.

MASK:MASKPRE:HSCALE? might return :MASK:MASKPRE:HSCALE 1, indicating that the current mask's nominal timing resolution is to 1 nanosecond per divisions.

MASK:MASKPRE:HTRIGPOS

This command sets or returns the nominal trigger position (pulse leading edge) used to draw the mask as a fraction of the display width. The query form of this command returns the nominal trigger position of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax
```
MASK:MASKPRE:HTRIGPOS <NR3>
MASK:MASKPRE:HTRIGPOS?
```

Related Commands MASK:MASKPRE:HSCAle

Arguments <NR3> is a floating point number in the range of 0.0 to 1.0 that sets the trigger points as a fraction of the display width, referenced from the left edge of the graticule. The number 0.0 represents the left edge.

Examples `MASK:MASKPRE:HTRIGPOS 0.6` sets the current mask nominal pulse leading edge position to the sixth horizontal graticule division.

`MASK:MASKPRE:HTRIGPOS?` might return `:MASK:MASKPRE:HTRIGPOS 0.8`, indicating that the trigger position is set to eight horizontal graticule divisions.

MASK:MASKPRE:PATTERNBITS

This command sets or returns the number of bits used for serial trigger for the User mask standard. For example, DS1 requires six bits: four leading zeros, a one, and a trailing zero. The query form of this command returns the serial bit value of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax
```
MASK:MASKPRE:PATTERNBITS <NR1>
MASK:MASKPRE:PATTERNBITS?
```

Related Commands MASK:MASKPRE:PRESAMPBITS

Arguments <NR1> is an integer that sets the number of bits.

Examples

`MASK:MASKPRE:PATTERNBITS 6` sets the current mask number of serial pattern bits to 6.

`MASK:MASKPRE:PATTERNBITS?` might return `:MASK:MASKPRE:PATTERNBITS 8`, indicating that the number of bits for serial trigger of the user mask standard is set to 8.

MASK:MASKPRE:PRESAMPBITS

This command sets or returns the number of bits before the (isolated one) pulse leading edge in the serial trigger pass/fail testing. For example, DS1 has four leading zeros. The query form of this command returns the presample bit value of the displayed mask. The set form of this command only affects the current (displayed) mask.

Group

Mask

Syntax

```
MASK:MASKPRE:PRESAMPBITS <NR1>
MASK:MASKPRE:PRESAMPBITS?
```

Related Commands

MASK:MASKPRE:PATTERNBITS

Arguments

`<NR1>` is an integer that sets the number of bits before the trigger pulse.

Examples

`MASK:MASKPRE:PRESAMPBITS 4` sets the current mask number of bits before the trigger pulse to 4.

`MASK:MASKPRE:PRESAMPBITS?` might return `:MASK:MASKPRE:PRESAMPBITS 8`, indicating that the presample bit value of the displayed mask is set to 8.

MASK:MASKPRE:RECOrdlength

This command sets or returns the nominal record length for pulse mask testing. The query form of this command returns the record length value of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group

Mask

Syntax

```
MASK:MASKPRE:RECOrdlength <NR1>
MASK:MASKPRE:RECOrdlength?
```

Arguments <NR1> is an integer number that sets the record length.

Examples MASK:MASKPRE:RECORDLENGTH 500 sets the current mask record length to 500 points.

MASK:MASKPRE:RECORDLENGTH? might return :MASK:MASKPRE:RECORDLENGTH 400, indicating that the nominal record length for pulse mask testing is set to 400.

MASK:MASKPRE:TRIGTOSAMP

This command sets or returns the time in seconds, from the (leading edge) trigger position to the pulse bit sampling position. The query form of this command returns the time value of the displayed mask. The set form of this command only affects the current (displayed) mask.

Group Mask

Syntax
```
MASK:MASKPRE:TRIGTOSAMP <NR3>
MASK:MASKPRE:TRIGTOSAMP?
```

Related Commands MASK:MASKPRE:WIDth

Arguments <NR3> is the floating point number that sets the time to the pulse bit sampling position.

Examples MASK:MASKPRE:TRIGTOSAMP 162E-9 sets the current mask time from the (leading edge) trigger position to the pulse bit sampling position to 162 nanoseconds.

MASK:MASKPRE:TRIGTOSAMP? might return :MASK:MASKPRE:TRIGTOSAMP 100, indicating that the time value of the displayed mask is set to 100 nanoseconds.

MASK:MASKPRE:VOFFSet

This command sets or returns the nominal vertical offset in volts, used to vertically offset the input channels. The query form of this command returns the offset value of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax

```
MASK:MASKPRE:VOFFSet <NR3>
MASK:MASKPRE:VOFFSet?
```

Related Commands MASK:MASKPRE:VPOS, MASK:MASKPRE:VSCAle

Arguments <NR3> is a floating point number that sets the nominal vertical offset in volts.

Examples MASK:MASKPRE:VOFFSET 0.0 sets the current mask nominal vertical offset to 0.0 volts.

MASK:MASKPRE:VOFFSET? might return :MASK:MASKPREVOFFSET 1.0000, indicating that the offset value of the displayed mask is set to 1.0 volts.

MASK:MASKPRE:VPOS

This command sets or returns the nominal vertical position, control in divisions, used to vertically position the input channels. The query form of this command returns the vertical position value of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax

```
MASK:MASKPRE:VPOS <NR3>
MASK:MASKPRE:VPOS?
```

Related Commands MASK:MASKPRE:VOFFSet, MASK:MASKPRE:VSCAle

Arguments <NR3> is a floating point number that sets the nominal vertical position control in divisions.

Examples MASK:MASKPRE:VPOS -0.8 sets the current mask nominal vertical position to -0.8 divisions.

MASK:MASKPRE:VPOS? might return :MASK:MASKPRE:VPOS 1.0000, indicating that the vertical position value of the displayed mask is set to 1.0 divisions.

MASK:MASKPRE:VSCAle

This command sets or returns the nominal vertical scale in volts per division, used to vertically scale the input channels. The query form of this command returns

the vertical scale value of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax
```
MASK:MASKPRE:VSCAle <NR3>
MASK:MASKPRE:VSCAle?
```

Related Commands MASK:MASKPRE:VOFFSet, MASK:MASKPRE:VPOS

Arguments <NR3> is a floating point number that sets the nominal bit width in seconds.

Examples MASK:MASKPRE:VSCALE 200E-3 sets the current mask nominal vertical scale to 200 millivolts per division.

MASK:MASKPRE:VSCALE? might return :MASK:MASKPRE:VSCALE 400.0000E-3, indicating that the vertical position value of the displayed mask is set to 400 millivolts per division.

MASK:MASKPRE:WIDth

This command sets or returns the nominal bit width in seconds. The query form of this command returns the bit width value of the displayed mask. The set form of this command affects only the current (displayed) mask.

Group Mask

Syntax
```
MASK:MASKPRE:WIDth <NR3>
MASK:MASKPRE:WIDth?
```

Related Commands MASK:MASKPRE:TRIGTOSAMP

Arguments <NR3> is a floating point number that sets the nominal bit width in seconds. This number is the time of one bit of data where bit width = 1 / (data rate of the signal).

Examples MASK:MASKPRE:WIDTH 648E-9 sets the current mask nominal bit width to 648 nanoseconds (1.544 Mbits/second).

MASK:MASKPRE:WIDTH? might return :MASK:MASKPRE:WIDTH 500.0000E-9, indicating that the bit width value of the displayed mask is set to 500 nanoseconds.

MASK:POLarity

This command sets or returns the input waveform polarity for the pass/fail test. It controls whether to test positive pulse, negative pulse, or both during pass/fail testing. This command only applies when MASK:TESt:STATE is on.

Group Mask

Syntax

```
MASK:POLarity {BOTh|NEGAtive|POSITIVe}
MASK:POLarity?
```

Related Commands MASK:TESt:STATE

Arguments

`BOTh` enables testing for both positive and negative pulses. The instrument tests positive pulses on the mask:source waveform until ½ of the waveform is tested. Then the instrument inverts the mask and performs the remaining tests.

`NEGAtive` enables testing on negative pulses.

`POSITIVe` enables testing on positive pulses. This is the default.

Examples

`MASK:POLARITY BOTH` sets pass/fail to test both positive (normal) and negative (inverted) waveforms.

`MASK:POLARITY?` might return `:MASK:POLARITY POSITIVE`, indicating that the input waveform polarity for the pass/fail test is set to positive.

MASK:SEG<m> (No Query Form)

This command deletes the specified mask segment from the current mask.

m is an integer that specifies the mask segment number to delete from the current mask.

Group Mask

Syntax

```
MASK:SEG<m> DELEte
```

Arguments

`DELETE` removes the specified mask segment from the mask.

Examples

`MASK:SEG3 DELETE` deletes mask segment three from the current mask.

MASK:SEG<m>:NR_Pt? (Query Only)

This query-only command returns the number of points that make up the specified mask segment of the current mask. Each mask point consists of an X-Y pair of coordinates.

m is an integer number that specifies a mask segment number of the current mask.

Group Mask

Syntax `MASK:SEG<m>:NR_Pt?`

Returns <NR1> is 0 to 50.

Examples `MASK:SEG2:NR_PT?` might return `:MASK:SEG2:NR_PT 5`.

MASK:SEG<m>:POINTS

This command sets or returns the X-Y user coordinates of all points in the current mask segment. The set form defines new points in the current mask, replacing any existing points in the current mask segment; it sets or returns the vertices for a particular segment in the current mask.

m is an integer that specifies the current mask segment number.

Group Mask

Syntax
```
MASK:SEG<m>:POINTS <NR3>,<NR3>[,<NR3>,<NR3>]
MASK:SEG<m>:POINTS?
```

Arguments <NR3> refers to the coordinates of one of the vertices in the Current mask. Each pair of numbers represents the horizontal and vertical coordinates of a mask segment vertex. The pairs must be listed in a counterclockwise order. If the vertical or horizontal scale or position is changed after this command and then the query form of this command is issued, the value returned from the instrument will not be the same. If just one pair is input, it is ignored and the current mask segment is marked as undefined. The default is no points in the current mask segment.

Examples `MASK:SEG2:POINTS -2.3E-9, 44E-3, -2.5E-9, 47E-3, 1.2E-9, 44E-3` defines the points in current mask segment 2.

MASK:SEG2:POINTS? might return :MASK:SEG2:POINTS 0.0000, 0.0000, indicating that the X-Y user coordinates of all points in the Current mask segment are set to 0,0.

MASK:SOUrce

This command sets or reports which source will be compared against the mask(s) when counting is turned on; it controls which trace to use in mask counting. It also affects mask autoset and how triggering is set up when you select the mask.

Group Mask

Syntax

```
MASK:SOUrce {CH[1-4]|MATH[1-4]|REF[1-4]}
MASK:SOUrce?
```

Arguments

CH<1-4> selects a channel waveform to be compared against the specified mask. The range for is 1 through 4.

MATH<1-4> selects a math waveform to be compared against the specified mask. The range for is 1 through 4.

REF<1-4> selects a reference waveform to be compared against the specified mask. The range is 1 through 4.

Examples

MASK:SOURCE CH1 enables CH1 to be compared against the mask. Since a timebase is not specified, the default, Main, is used.

MASK:SOURCE? might return :MASK:SOURCE CH1,MAIN, indicating that the Channel 1 waveform will be compared against the mask using the Main timebase.

MASK:STANdard

This command deletes the existing mask (if any) and sets the selected standard mask. If MASK:COUNt:STATE is ON, mask counting starts immediately. The query form of this command returns the current mask standard.

NOTE. *There are several available mask standards that require a higher bandwidth than permitted for some instruments. In such cases, a warning is posted to the standard event system upon selection of any of these standards. To ensure that the mask you select is fully supported by the instrument, always check the event status register after selecting a mask standard.*

The following warning event is posted if the mask exceeds the instrument bandwidth: 2318,“Consider system bandwidth when testing at this bit rate.”

Conditions This command is only valid for instruments with Option MTH or MTM installed.

Group Mask

Syntax

```
MASK:STANdard {ATARXG1|ATARXG2|ATARXG3|ATATXG1|ATATXG2|
ATATXG3|CLOCKCoax|CLOCKSymmetrical|D1|D2
|DS0Contra|DS0Double|
DS0Single|DS0Timing|DS1|DS1A|DS1C|DS2|DS2RATECoax|
DS2RATESymmetrical|DS3|DS4NA|DS4NA_Max|E1Coax
|E1Symmetrical|
E2|E3|E4_1|E4_0|ENET1000BCX_ATP2|ENET1000BCX_ATP3|
ENET1000BCX_NTP2|ENET100FX|ENET100STP|ENET100UTP |ENET1250|
ENETXAUI_Far|ENETXAUI_Near|FC1063|FC1063Draft|FC1063E|
FC1063E_ABR|FC1063E_ABT|FC1063E_ADR|FC1063E_ADT
|FC1063E_AGR|
FC1063E_AGT|FC1063E_NBT|FC1063E_NDT|FC1063E_NGT|FC133|
FC133E|FC2125|FC2125E_ABR|FC2125E_ABT|FC2125E_ADR|FC2125E_ADT|
FC2125E_AGR|FC2125E_AGT|FC2125E_NBT|FC2125E_NDT
|FC2125E_NGT|FC266|FC266E|FC4250E_ABR|FC4250E_ABT
|FC4250E_ADR|FC4250E_ADT|FC4250E_AGR
|FC4250E_AGT|FC4250E_NBT|FC4250E_NDT
|FC4250E_NGT|FC531|FC531E|FST1|FST2|FST3
|FST4|FST5|FST6|FW1394BS1600B|
FW1394BS1600BT1|FW1394BS1600BT2|FW1394BS400B
|FW1394BS400BT1|
FW1394BS400BT2|FW1394BS800B|FW1394BS800BT1|FW1394BS800BT2|
G703DS1|G703DS3|HST1|HST2|HST3|HST4|HST5|HST6|INF2_5G|
INF2_5GE|NONe|OC1|OC12|OC3|OC48|OC48_FEC|PCIEXPRESS_Rcv|
PCIEXPRESS_Xmit|RATE32Mbit|RATE97Mbit|RIO_DRV1G
|RIO_DRV1_5G|
RIO_DRV2G|RIO_DRV500M|RIO_DRV500M|RIO_DRV750M|RIO_EDRV1G|
RIO_EDRV1_5G|RIO_EDRV2G|RIO_EDRV500M|RIO_EDRV500M|
RIO_EDRV750M|RIO_RCV1G|RIO_RCV1_5G|RIO_RCV2G
|RIO_RCV500M| RIO_RCV500M|RIO_RCV750M|RIO_SERIAL_1G
|RIO_SERIAL_2G|RIO_SERIAL_3G|
SFI5_RCVBCLK2|SFI5_RCVBCLK3|SFI5_RCVBDATA2 |SFI5_RCVBDATA3|
SFI5_RCVDCLK2|SFI5_RCVDCLK3|SFI5_RCVDDATA2|SFI5_RCVDDATA3|
SFI5_XMITACLK2|SFI5_XMITACLK3|SFI5_XMITADATA2|SFI5_XMITADATA3|
SFI5_XMITCCLK2|SFI5_XMITCCLK3|SFI5_XMITCDATA2
|SFI5_XMITCDATA3|
STM0_0|STM0_1|STM0_HDBX|STM1E_1|STM1E_0|STS1Eye
|STS1Pulse| STS3|STS3_Max|TFI5_2|TFI5_3|USERMask|VIDEO270
|VIDEO292M|VIDEO360|VSROC192|SAS1_5_IR|SAS1_5_CR|SAS1_5_XR
|SAS1_5_IR_AASJ|SAS1_5_CR_AASJ|SAS1_5_XR_AASJ
|SAS1_5_SATA|SAS3_0_IR|SAS3_0_CR|SAS3_0_XR
|SAS3_0_IR_AASJ|SAS3_0_CR_AASJ|SAS3_0_XR_AASJ |SAS3_0_SATA}
```

MASK:STANdard?

Related Commands ACQuire:MODe

Arguments

ATARXG1 (Serial ATA, G1 Rx 1.5 Gb/s)

ATARXG2 (Serial ATA, G2, Rx, 3.0 Gb/s)

ATARXG3 (Serial ATA, G3, Rx)

ATATXG1 (Serial ATA, G1 Tx, 1.5 Gb/s)

ATATXG2 (Serial ATA, G2 Tx, 3.0 Gb/s)

H ATATXG3 (Serial ATA, G3 Tx)

CLOCKCoax

CLOCKSymmetrical

D1

D2

DS0Contra (ITU-T, G703 (10/98), 64 kb/s)

DS0Double (ITU-T, G703 (10/98), 64 kb/s)

DS0Single (ITU-T, G703 (10/98), 64 kb/s)

DS0Timing (ITU-T, G703 (10/98), 64 kb/s)

DS1 (ANSI T1.102-1993 (R1999), DS1, 1.544 Mb/s)

DS1A (ANSI T1.102-1993 (R1999), DS1A, 2.048 Mb/s)

DS1C (ANSI T1.102-1993 (R1999), DS1C, 3.152 Mb/s)

DS2 (ANSI T1.102-1993 (R1999), DS2, 6.312 Mb/s)

DS2RATECoax (ITU-T, G703 (10/98), D2 Rate Coax, 6.312 Mb/s)

DS2RATESymmetrical (ITU-T, G703 (10/98), D2 Rate Sym, 6.312 Mb/s)

DS3 (ANSI T1.102-1993 (R1999), DS3, 44.736 Mb/s)

DS4NA (ANSI T1.102-1993 (R1999), DS4NA, 139.26 Mb/s)

DS4NA_Max (ANSI T1.102-1993 (R1999), DSNA Max Output, 139.26 Mb

E1Coax (ITU-T, G703 (10/98), E1 Coax Pair, 2.048 Mb/s)

E1Symmetrical (ITU-T, G703 (10/98), E1 Sym Pair, 2.048 Mb/s)

E2 (ITU-T, G703 (10/98), E2, 8.448 Mb/s)

E3 (ITU-T, G703 (10/98), E3, 34.368 Mb/s)

E4_0 (ITU-T, G703 (10/98), E4 Binary 0)

E4_1 (ITU-T, G703 (10/98), E4 Binary 1)

ENET100FX

ENET100STP (IEEE Std 802.3 and ANSI X3.263-1995, 100 Base-Tx, STP, 125 Mb/s)

ENET100UTP (IEEE Std 802.3 and ANSI X3.263-1995, 100 Base-Tx, UTP, 125 Mb/s)

ENET1000BCX_ATP2 (1000B-CX Abs, TP2, 1.25 Gb/s)

ENET1000BCX_ATP3 (1000B-CX Abs, TP3, 1.25 Gb/s)

ENET1000BCX_NTP2 (1000B-CX Norm, TP2, 1.25 Gb/s)

ENET1250 (IEEE Std 802.3 and ANSI X3.263-1995, GB Ethernet, 1.25 Gb/s)

ENETXAUI_FAR (10 Gigabit Attachment Unit Interface (XAUI), Far, 3.125 Gb/s)

ENETXAUI_Near (10 Gigabit Attachment Unit Interface (XAUI), Near, 3,125 Gb/s)

FC133 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 132.8 Mb/s)

FC133E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical 132.8 Mb/s)

FC266 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 265.6 Mb/s)

FC266E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical, 132.8 Mb/s)

FC531 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 531.2 Mb/s)

FC531E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical, 531.2 Mb/s)

FC1063 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 1.065 Gb/s)

FC1063E (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Electrical, 1.0625 Gb/s)

FC1063E_ABT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Transm)

FC1063E_ADT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Transm)

FC1063E_AGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Transm)

FC1063E_NBT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Beta, Transm)

FC1063E_NDT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Delta, Transm)

FC1063E_NGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Gamma, Transm)

FC1063E_ABR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Recv)

FC1063E_ADR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Delta, Recv)

FC1063E_AGR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Recv)

FC1063Draft (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, Draft Rev 11)

FC2125 (ANSI X3.230-1999 NCITS 1235D/Rev 11, Optical, 2.125)

FC2125E_ABT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Transm)

FC2125E_ADT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Delta, Transm)

FC2125E_AGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Transm)

FC2125E_NBT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Beta, Transm)

FC2125E_NDT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Delta, Transm)

FC2125E_NGT (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Norm, Gamma, Transm)

FC2125E_ABR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Beta, Recv)

FC2125E_ADR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Delta, Recv)

FC2125E_AGR (ANS1 X3.230-1999 NCITS 1235D/Rev 11, Abs, Gamma, Recv)

FC4250E_ABT (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Abs, Beta, Transm)

FC4250E_ADT (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Abs, Delta, Transm)

FC4250E_AGT (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Abs, Gamma, Transm)

FC4250E_NBT (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Norm, Beta, Transm)

FC4250E_NDT (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Norm, Delta, Transm)

FC4250E_NGT (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Norm, Gamma, Transm)

FC4250E_ABR (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Abs, Beta, Recv)

FC4250E_ADR (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Abs, Delta, Recv)

FC4250E_AGR (ANS1 X3.230-1999 NCITS 1235D/Rev 4.0, Abs, Gamma, Recv)

FST1 (USB, FS:T1, 12 Mb/s)

FST2 (USB, FS:T2, 12 Mb/s)

FST3 (USB, FS:T3, 12 Mb/s)

FST4 (USB, FS:T4: 12 Mb/s)

FST5 (USB, FS:T5, 12 Mb/s)

FST6 (USB, FS:T6, 12 Mb/s)

FW1394BS400B (IEEE 1394b, S400 Optical, 491.5 Mb/s)

FW1394BS400BT1 (IEEE 1394b, S400b T1, 491.5 Mb/s)

FW1394BS400BT2 (IEEE 1394b, S400b T2, 491.5 Mb/s)

FW1394BS800B (IEEE 1394b, S800 Optical, 988.0 Mb/s)

FW1394BS800BT1 (IEEE 1394b, S800b T1, 983.0 Mb/s)

FW1394BS800BT2 (IEEE 1394b, S800b T2, 983.0 Mb/s)

FW1394BS1600B (IEEE 1394b, S1600 Optical, 1.966 Gb/s)

FW1394BS1600BT1 (IEEE 1394b, S1600b T1, 1.966 Gb/s)

FW1394BS1600BT2 (IEEE 1394b, S1600b T2, 1.966 Gb/s)

G703D1 (ITU-T, G703 (10/98), DS1 Rate, 1.544 Mb/s)

G703DS3 (ITU-T, G703 (10/98)

HST1 (USB, HS:T1, 480 Mb/s) G703DS3 (ITU-T, G703 (10/98)

HST2 (USB, HS:T2, 480 Mb/s) G703DS3 (ITU-T, G703 (10/98)

HST3 (USB, HS:T3, 480 Mb/s)

HST4 (USB, HS:T4, 480 Mb/s)

HST5 (USB, HS:T5, 480 Mb/s)

HST6 (USB, HS:T6, 480 Mb/s)

INF2_5G (InfiniBand, IBTA Spec 1.0a, 2.5 Optical, 2.5 Gb/s)

INF2_5GE (InfiniBand, IBTA Spec 1.0a, 2.5 Electrical, 2.5 Gb/s)

NONe

OC1 (GR 253-CORE Issue 3 9/21/2000 OC1/STM0, 51.84 Mb/s)

OC3 (GR 253-CORE Issue 3 9/21/2000 OC1/STM1, 155.52, Mb/s)

OC12 (GR 253-CORE Issue 3 9/21/2000 OC1/STM4, 622.08 Mb/s)

OC48 (GR 253-CORE Issue 3 9/21/2000 OC1/STM16, 2.4883 Gb/s

OC48_FEC (Forward Error Correction - CSA8000 mask, 2.666 Gb/s)

PCIEXPRESS_Xmit (PCI Express Transmitter, 2.5 Gb/s)

PCIEXPRESS_Rcv (PCI Express Receiver, 2.5 Gb/s)

RATE32Mbit (ITU-T, G703 (10/98), 32.064 Mb/s)

RATE97Mbit (ITU-T, G703 (10/98), 97 Mbit, 97.728 Mb/s)

RIO_DRV1G (Rapid IO Driver, 1 Gb/s)

RIO_DRV1_5G (Rapid IO Driver, 5 Gb/s)

RIO_DRV2G (Rapid IO Driver, 2 Gb/s)

RIO_DRV500M (Rapid IO Driver, 500 Mb/s)

RIO_DRV750M (Rapid IO Driver, 750 Mb/s)

RIO_EDRV1G (Rapid IO Extended Driver, 1 Gb/s)

RIO_EDRV1_5G (Rapid IO Extended Driver, 1.5 Gb/s)

RIO_EDRV2G (Rapid IO Extended Driver, 2 Gb/s)

RIO_EDRV500M (Rapid IO Extended Driver, 500 Mb/s)

RIO_EDRV750M (Rapid IO Extended Driver, 750 Mb/s)

RIO_RCV500M (Rapid IO Receiver, 500 Mb/s)

RIO_RCV750M (Rapid IO Receiver, 750 Mb/s)

RIO_RCV1G (Rapid IO Receiver, 1 Gb/s)

RIO_RCV1_5G (Rapid IO Receiver, 1.5 Gb/s)

RIO_RCV2G (Rapid IO Receiver, 2 Gb/s)

RIO_SERIAL_1G (Rapid IO Serial, 1.25 Gb/s)

RIO_SERIAL_2G (Rapid IO Serial, 2.5 Gb/s)

RIO_SERIAL_3G (Rapid IO Serial, 3.25 Gb/s)

SFI5_XMITADATA2 (SFI15 Transmit: Test Point A Data Signal 2, 2.488 Gb/s)

SFI5_XMITCDATA2 (SFI15 Transmit: Test Point C Data Signal 2, 2.488 Gb/s)

SFI5_XMITACLK2 (SFI15 Transmit: Test Point A Clock Signal 2, 2.488 Gb/s)

SFI5_XMITCCLK2 (SFI15 Transmit: Test Point C Clock Signal 2, 2.488 Gb/s)

SFI5_RCVBDATA2 (SFI15 Receive: Test Point B Data Signal 2, 2.488 Gb/s)

SFI5_RCVDDATA2 (SFI15 Receive: Test Point D Data Signal 2, 2.488 Gb/s)

SFI5_RCVBCLK2 (SFI15 Receive: Test Point B Clock Signal 2, 2.488 Gb/s)

SFI5_RCVDCLK2 (SFI15 Receive: Test Point D Clock Signal 2, 2.488 Gb/s)

SFI5_XMITADATA3 (SFI15 Transmit: Test Point A Data Signal 3, 3.125 Gb/s)

SFI5_XMITCDATA3 (SFI15 Transmit: Test Point C Data Signal 3, 3.125 Gb/s)

SFI5_XMITACLK3 (SFI15 Transmit: Test Point A Clock Signal 3, 3.125 Gb/s)

SFI5_XMITCCLK3 (SFI15 Transmit: Test Point C Clock Signal 3, 3.125 Gb/s)

SFI5_RCVBDATA3 (SFI15 Receive: Test Point B Data Signal 3, 3.125 Gb/s)

SFI5_RCVDDATA3 (SFI15 Receive: Test Point D Data Signal 3, 3.125 Gb/s)

SFI5_RCVBCLK3 (SFI15 Receive: Test Point B Clock Signal 3, 3.125 Gb/s)

SFI5_RCVDCLK3 (SFI15 Receive: Test Point D Clock Signal 3, 3.125 Gb/s

STM0_0 (ITU-T, G703 (10/98), STM1E Binary 0)

STM0_1 (ITU-T, G703 (10/98), STM1E Binary 1)

STM0_HDBX

STS1Eye (ANSI T1.102-1993 (R1999), STS-1 Eye, 51.84 Mb/s)

STS1Pulse (ANSI T1.102-1993 (R1999), STS-1 Pulse, 51.84 Mb/s)

STS3 (ANSI T1.102-1993 (R1999), STS-3, 155.52 Mb/s)

STS3_Max (ANSI T1.102-1993 (R1999), STS-3 Max Output, 155.52 Mb/s)

TFI15_2 (TFI-5, 2.488 Gb/s)

TFI5_3 (TFI-5, 3.1104 Gb/s)

USERMask

VIDEO270

VIDEO292M

VIDEO360

VSROC192 (VSR OC192/STM64, 1.24416 Gb/s)

Examples

MASK:STANDARD E1COAX selects the E1 Coax standard mask.

MASK:STANDARD? might return :MASK:STANDARD E1COAX, indicating that the E1 Coax mask is selected as the current mask standard.

MASK:STOPOnviolation

This command sets or returns the stop on violation setting. When enabled, this command stops waveform acquisitions on the first occurrence of a waveform violation. MASK:DISplay must be ON for the instrument to detect violations.

Group

Mask

Syntax

```
MASK:STOPOnviolation {<NR1>|OFF|ON}
MASK:STOPOnviolation?
```

Arguments

<NR1> = 0 turns off stop on violation; any other value turns on stop on violation.

OFF turns off mask test stop on violation.

ON stops waveform acquisition on the first occurrence of a mask violation.

Examples

`MASK:STOPONVIOLATION ON` sets the instrument to stop waveform acquisition on the first occurrence of a waveform violation.

`MASK:STOPONVIOLATION?` might return `:MASK:STOPONVIOLATION ON`, indicating that the stops waveform acquisitions on the first occurrence of a waveform violation is enabled.

MASK:TESt:AUX:COMPLetion

This command sets or returns the test:aux on pass/fail test completion mode. When enabled, this command causes the instrument to provide a TTL signal at the Aux:Out port when the mask pass/fail status changes to ‘Failing'.

Group

Mask

Syntax

```
MASK:TESt:AUX:COMPLetion {<NR1>|OFF|ON}
MASK:TESt:AUX:COMPLetion?
```

Arguments

`<NR1>` = 0 disables the mask:test:aux:completion function; any other value enables it.

`OFF` turns off the pass/fail mask:test:aux on completion.

`ON` turns on the pass/fail mask:test:aux on completion.

Examples

`MASK:TEST:AUX:COMPLETION ON` turns on the pass/fail test on completion.

`MASK:TEST:AUX:COMPLETION?` might return `:MASK:TEST:AUX:COMPLETION ON`, indicating that the `test:aux` on pass/fail test completion mode is turned on.

MASK:TESt:AUX:FAILure

This command sets or returns the `test:aux` status on pass/fail test failure mode. When enabled, this command causes the instrument to provide a TTL signal at the AuxOut port when the pass/fail status changes to ‘Failing'.

Group

Mask

Syntax

```
MASK:TESt:AUX:FAILure {<NR1>|OFF|ON}
MASK:TESt:AUX:FAILure?
```

Arguments

<NR1> = 0 disables the mask:test:aux:failure function; any other value enables it.

OFF turns off the pass/fail mask:test:aux on failure. This is the default.

ON turns on the pass/fail mask:test:aux on failure.

Examples

MASK:TEST:AUX:FAILURE ON turns on the pass/fail test on failure.

MASK:TEST:AUX:FAILURE? might return :MASK:TEST:AUX:FAILURE ON, indicating that the pass/fail mask:test:aux on failure is turned on.

MASK:TESt:BEEP:COMPLetion

This command sets or returns the beep on pass/fail test completion mode. When enabled, this command causes the instrument to emit a tone when the mask pass/fail test completes.

Group

Mask

Syntax

```
MASK:TESt:BEEP:COMPLetion {<NR1>|OFF|ON}
MASK:TESt:BEEP:COMPLetion?
```

Arguments

<NR1> = 0 turns off the pass/fail beep on completion, any other integer turns on the pass/fail beep on completion.

OFF turns off the pass/fail beep on completion.

ON turns on the pass/fail beep on completion.

Examples

MASK:TEST:BEEP:COMPLETION ON turns on pass/fail beep on completion.

MASK:TEST:BEEP:COMPLETION? might return :MASK:TEST:BEEP:COMPLETION ON, indicating that the pass/fail beep on completion is turned on.

MASK:TESt:BEEP:FAILure

This command sets or returns the beep status on pass/fail test failure mode. When enabled, this command causes the instrument to emit a tone when the pass/fail status changes to ‘Failing'.

Group

Mask

Syntax

```
MASK:TESt:BEEP:FAILure {<NR1>|OFF|ON}
MASK:TESt:BEEP:FAILure?
```

Arguments

<NR1> = 0 turns off the pass/fail beep on failure, and any other integer turns on the pass/fail beep on failure.

OFF turns off the pass/fail beep on failure. This is the default.

ON turns on the pass/fail beep on failure.

Examples

MASK:TEST:BEEP:FAILURE OFF turns off the pass/fail beep on failure.

MASK:TEST:BEEP:FAILURE? might return :MASK:TEST:BEEP:FAILURE ON, indicating that the pass/fail beep on failure is turned on.

MASK:TESt:DELay

This command sets or returns the amount of time in seconds, to wait after starting pass/fail testing to evaluate waveforms. This is useful if the test system requires 'settling' time before evaluating waveforms.

Group

Mask

Syntax

```
MASK:TESt:DELay <NR3>
MASK:TESt:DELay?
```

Arguments

<NR3> is a floating point number that specifies the amount of time in seconds, to delay the start of the pass/fail test.

Examples

MASK:TEST:DELAY 2.5 delays waveform evaluation by 2.5 seconds after the start of pass/fail testing.

MASK:TEST:DELAY? might return :MASK:TEST:DELAY 5, indicating that the amount of time to wait after starting pass/fail testing to evaluate waveforms is set to 5 seconds.

MASK:TESt:HARDCopy

This command sets or returns the hard copy in pass/fail failure mode. When enabled, this command causes the instrument to generate a screen hard copy, to the default printer using the current instrument hard copy settings, as soon as the pass/fail status changes to 'Failing'.

Group Mask

Syntax
```
MASK:TESt:HARDCopy {<NR1>|OFF|ON}
MASK:TESt:HARDCopy?
```

Arguments <NR1> = 0 turns off generate hard copy on failure, and any other value turns on generate hard copy on failure.

OFF turns off generate hard copy on failure.

ON turns on generate hard copy on failure.

Examples MASK:TEST:HARDCOPY ON turns on generate hard copy on failure.

MASK:TEST:HARDCOPY? might return :MASK:TEST:HARDCOPY ON, indicating that the generate hard copy on failure is turned on.

MASK:TESt:LOG:FAILure

This command sets or returns the log status on pass/fail test failure mode. When enabled, this command causes the instrument to log the current date and time to a file when the pass/fail status changes to 'Failing'. The file name is determined from the current date and time. The user cannot change this name.

Group Mask

Syntax
```
MASK:TESt:LOG:FAILure {<NR1>|OFF|ON}
MASK:TESt:LOG:FAILure?
```

Related Commands MASK:TESt:SAVEWFM:FILEName

Arguments <NR1> = 0 turns off the pass/fail log on failure, and any other integer turns on the pass/fail log on failure.

OFF turns off the pass/fail log on failure. This is the default.

ON turns on the pass/fail log on failure.

Examples MASK:TEST:LOG:FAILURE OFF turns off the pass/fail log on failure.

MASK:TEST:LOG:FAILURE? might return :MASK:TEST:LOG:FAILURE ON, indicating that the pass/fail log on failure is turned on.

MASK:TESt:REPeat

This command sets or returns the repeat pass/fail test on completion mode. When enabled, this command repeats the pass/fail test cycle at the completion of the previous test cycle.

Group Mask

Syntax

```
MASK:TESt:REPeat {<NR1>|OFF|ON}
MASK:TESt:REPeat?
```

Related Commands MASK:COUNt:VIOLATIONS?

Arguments

<NR1> = 0 turns off repeat on completion, and any other integer turns on repeat on completion.

OFF turns off repeat on completion.

ON turns on repeat on completion.

Examples

MASK:TEST:REPEAT ON causes the instrument to continuously repeat the pass/fail test.

MASK:TEST:REPEAT? might return :MASK:TEST:REPEAT ON, indicating that the repeat pass/fail test on completion is turned on.

MASK:TESt:SAMple

This command sets or returns the minimum number of waveform database (WfmDB) points the instrument can acquire before it stops a single sequence acquisition or stops running a mask test.

Hint: a bigger sample size can allow a greater throughput (more waveforms to get acquired at a time). However, a very large sample size can slow down the display update. So to speed up the display rate, consider reducing the sample size.

This command works the same as the ACQuire:NUMSAMples command.

Group Mask

Syntax

```
MASK:TESt:SAMple {<NR1>}
MASK:TESt:SAMple?
```

Arguments <NR1> the number of points to sample.

Examples MASK:TEST:SAMPLE 100000 sets the instrument to acquire 100,000 WfmDB points.

MASK:TEST:SAMPLE? might return :MASK:TEST:SAMPLE 100000, indicating that the repeat pass/fail test on completion is set to 100,000 WfmDB points.

MASK:TESt:SAMple:THReshold

This command sets or returns the minimum number of hits in mask regions needed to cause the pass/fail status to change from PASSING to FAILING. This affects the mask test when the instrument is acquiring in waveform database (WfmDB) mode.

Group Mask

Syntax
```
MASK:TESt:SAMple:THReshold {<NR1>}
MASK:TESt:SAMple:THReshold?
```

Arguments <NR1> is the number of hits that can happen when the test fails.

Examples MASK:TEST:SAMPLE:THRESHOLD 100 sets the test to fail when 100 or more hits happen.

MASK:TEST:SAMPLE:THRESHOLD? might return :MASK:TEST:THRESHOLD 1000, indicating that the repeat pass/fail test on completion is set to 1000 WfmDB points.

MASK:TESt:SAVEWFM

This command sets or returns the save waveform to file on failure setting. When enabled, this command copies waveform data from active channels to files when the pass/fail test status changes to 'Failing'.

NOTE. *This instrument saves files only on the first violation that causes the pass/fail test to fail. To generate a waveform file for every violation occurrence, set MASK:TESt:REPeat to ON and MASK:TESt:WAVEform to 1. Use this technique only when the expected number of violations is small and the violation occurs intermittently.*

Group Mask

Syntax
```
MASK:TESt:SAVEWFM {<NR1>|OFF|ON}
MASK:TESt:SAVEWFM?
```

Arguments <NR1> = 0 turns off copying all active channel waveforms to files, and any other integer turns on copying all active channel waveforms to files.

OFF turns off copying all active channel waveforms to files.

ON turns on copying all active channel waveforms to files.

Examples MASK:TEST:SAVEWFM ON sets the instrument to save the failing waveform data to files.

MASK:TEST:SAVEWFM? might return :MASK:TEST:SAVEWFM 1000, indicating that the repeat pass/fail test on completion is set to 1000 WfmDB points.

MASK:TESt:SAVEWFM:FILEName

This command sets or returns the name of the directory and file to use with the MASK:TESt:SAVEWFM command. It defines the directory the files will be put in (name comes from date and time).

Group Mask

Syntax
```
MASK:TESt:SAVEWFM:FILEName <QString>
MASK:TESt:SAVEWFM:FILEName?
```

Related Commands MASK:TESt:SAVEWFM, MASK:TESt:LOG:FAILure

Arguments <QString> is a string representing the name of the file to save waveform data to.

Examples MASK:TEST:SAVEWFM:FILENAME "TEST" sets the file “TEST” as the file to save waveform data to on failure.

MASK:TEST:SAVEWFM:FILENAME? might return :MASK:TEST:SAVEWFM:FILENAME “C:/TekScope/Waveforms”, indicating that the directory “C:/TekScope/Waveforms” the files will be put in.

MASK:TESt:SRQ:COMPLetion

This command sets or returns an SRQ on pass/fail test completion mode. When enabled, this command causes the instrument to send a GPIB SRQ command when the mask pass/fail test completes.

Group Mask

Syntax

```
MASK:TESt:SRQ:COMPLetion {<NR1>|OFF|ON}
MASK:TESt:SRQ:COMPLetion?
```

Arguments

<NR1> = 0 turns off the pass/fail SR on completion, any other integer turns on the pass/fail SRQ on completion.

OFF turns off the pass/fail SRQ on completion.

ON turns on the pass/fail SRQ on completion.

Examples

MASK:TEST:SRQ:COMPLETION ON turns on pass/fail SRQ on completion.

MASK:TEST:SRQ:COMPLETION? might return :MASK:TEST:SRQ:COMPLETION 1, indicating that the pass/fail SRQ on completion is turned on.

MASK:TESt:SRQ:FAILure

This command sets or returns the SRQ status on pass/fail test failure mode. When enabled, this command causes the instrument to send a GPIB SRQ command when the pass/fail status changes to ‘Failing'. In other words, it sends an SRQ when a failure occurs during the mask pass/fail test.

Group Mask

Syntax

```
MASK:TESt:SRQ:FAILure {<NR1>|OFF|ON}
MASK:TESt:SRQ:FAILure?
```

Arguments

<NR1> = 0 turns off the pass/fail SRQ on failure, and any other integer turns on the pass/fail SRQ on failure.

OFF turns off the pass/fail SRQ on failure. This is the default.

ON turns on the pass/fail SRQ on failure.

Examples `MASK:TEST:SRQ:FAILURE OFF` turns off the pass/fail SRQ on failure.

`MASK:TEST:SRQ:FAILURE?` might return `:MASK:TEST:SRQ:FAILURE 1`, indicating that the pass/fail SRQ on failure is turned on.

MASK:TESt:STATE

This command sets or returns the state of mask pass/fail testing. When enabled, this command resets the pass/fail status information and starts pass/fail mask testing. This command also generates an 'Operation Complete' message when pass/fail testing completes. Refer to BUSY? and *OPC commands for information on synchronizing a controller with pass/fail testing.

Group Mask

Syntax
```
MASK:TESt:STATE {<NR1>|OFF|ON}
MASK:TESt:STATE?
```

Arguments `<NR1>` = 0 turns off mask pass/fail testing; any other integer turns on mask pass/fail testing.

`OFF` turns off mask pass/fail testing.

`ON` turns on mask pass/fail testing.

Examples `MASK:TEST:STATE ON` turns on mask pass/fail testing.

`MASK:TEST:STATE?` might return `:MASK:TEST:STATE 0`, indicating that the mask pass/fail testing is turned off.

MASK:TESt:STATUS? (Query Only)

This query-only command returns the pass/fail test status. This command returns one of: OFF, DELAY, PASSING, FAILING, PASSED, FAILED, and VIOLATION. In other words, it indicates the result of the pass/fail test. When the violation count exceeds the violation threshold, the status changes from Passing to Failed.

Group Mask

Syntax `MASK:TESt:STATUS?`

Examples `MASK:TEST:STATUS?` might return `:MASK:TEST:STATUS PASSING` indicates pass test status.

MASK:TESt:STOP:FAILure

This command sets or returns the stop status on pass/fail test failure mode. When enabled, this command causes the instrument to stop acquiring data when the pass/fail status changes to 'Failing'. Repeat-on-completion mode has no effect.

Group Mask

Syntax

```
MASK:TESt:STOP:FAILure {<NR1>|OFF|ON}
MASK:TESt:STOP:FAILure?
```

Arguments `<NR1>` = 0 turns off the pass/fail stop on failure, and any other integer turns on the pass/fail SRQ on failure.

`OFF` turns off the pass/fail stop on failure. This is the default.

`ON` turns on the pass/fail stop on failure.

Examples `MASK:TEST:STOP:FAILURE OFF` turns off the pass/fail stop on failure.

`MASK:TEST:STOP:FAILURE?` might return `:MASK:TEST:STOP:FAILURE 0`, indicating that the mask pass/fail testing is turned off.

MASK:TESt:THReshold

This command sets or returns the number of failed tested waveforms needed in each pass/fail test to cause the pass/fail status to change from 'Passing' to 'Failing'.

Group Mask

Syntax

```
MASK:TESt:THReshold <NR1>
MASK:TESt:THReshold?
```

Arguments `<NR1>` is an integer that represents the number of tested waveform violations occurring in each pass/fail test needed to change the pass/fail status to 'Failing'. This means that if the threshold is set to the default of 1 then 1 failure would cause a test failure. If the threshold is set to 2 then 2 failures would cause a test failure.

Examples

`MASK:TEST:THRESHOLD 20` sets the number of failed tested waveforms to 20 that is needed in each pass/fail test to cause the pass/fail status to change from 'Passing' to 'Failing'.

`MASK:TEST:THRESHOLD?` might return `:MASK:TEST:THRESHOLD 32`, indicating that the 32 failures would cause a test failure.

MASK:TESt:WAVEform

This command sets or returns the number of waveforms to test during pass/fail testing. If Averaging is on, this is the number of averaged waveforms to test during pass/fail testing. It specifies the number of waveforms to be acquired in counting the mask hits in the pass/fail test. The default is 20.

Group

Mask

Syntax

```
MASK:TESt:WAVEform <NR1>
MASK:TESt:WAVEform?
```

Arguments

`<NR1>` is an integer that sets the number of tested (sampled or averaged) waveforms to test for each pass/fail test.

Examples

`MASK:TEST:WAVEFORM 20` sets the number of failures that would cause a test failure to 20.

`MASK:TEST:WAVEFORM?` might return

`:MASK:TEST:WAVEFORM 50`, indicating that the 50 failures would cause a test failure.

MASK:USER:AMPlitude

This command sets or returns the current mask's nominal pulse amplitude in volts. The query form of this command returns the nominal pulse amplitude of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group

Mask

Syntax

```
MASK:USER:AMPlitude <NR3>
MASK:USER:AMPlitude?
```

Related Commands MASK:USER:VSCAle

Arguments <NR3> is a floating number that sets the nominal pulse amplitude in volts.

Examples MASK:USER:AMPLITUDE 2.4 sets the user mask nominal pulse amplitude to 2.4 volts.

MASK:USER:AMPLITUDE? might return

:MASK:USER:AMPLITUDE 100.0000E-3, indicating that the user mask nominal pulse amplitude to set to 0.1 volts.

MASK:USER:BITRate

This command sets or returns the bit rate for the user mask.

Group Mask

Syntax
```
MASK:USER:BITRate <NR1>
MASK:USER:BITRate?
```

Arguments <NR1> is a number that sets the bit rate of the user mask in bits per second.

Examples MASK:USER:BITRATE 156E6 sets the bit rate.

MASK:USER:BITRATE? might return :MASK:USER:BITRATE 1560000, indicating that the bit rate of the user mask is set to 1560000 bits per second.

MASK:USER:HSCAle

This command sets or returns the nominal timing resolution used to draw the mask in time/division. The query form of this command returns the nominal timing resolution of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group Mask

Syntax
```
MASK:USER:HSCAle <NR3>
MASK:USER:HSCAle?
```

Related Commands MASK:USER:HTRIGPOS

Arguments <NR3> is a floating point number that sets the mask drawing timing resolution.

Examples MASK:USER:HSCALE 200E-9 sets the nominal timing resolution used to draw the mask to 200 nanoseconds per division.

MASK:USER:HSCALE? might return :MASK:USER:HSCALE 40.0000E-9, indicating that the nominal timing resolution used to draw the mask pulse shape is set to 40 nanoseconds per division.

MASK:USER:HTRIGPOS

This command sets or returns the nominal trigger position (pulse leading edge) used to draw the mask as a fraction of the display width. The query form of this command returns the nominal trigger position of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group Mask

Syntax
```
MASK:USER:HTRIGPOS <NR3>
MASK:USER:HTRIGPOS?
```

Related Commands MASK:USER:HSCAle

Arguments <NR3> is a floating point number in the range of 0.0 to 1.0 that sets the trigger points as a fraction of the display width, referenced from the left edge of the graticule.

0.0 represents the left edge.

Examples MASK:USER:HTRIGPOS 0.6 sets the nominal trigger position of the displayed mask to 0.6.

MASK:USER:HTRIGPOS? might return :MASK:USER:HTRIGPOS 500.0000E-3, indicating that the nominal trigger position of the displayed mask is set to 0.5.

MASK:USER:LABel

This command sets or queries the label assigned to a user mask. This label is displayed in the upper left-hand corner of the graticule when a user mask is selected.

Group Mask

Syntax
```
MASK:USER:LABel <QString>
MASK:USER:LABel?
```

Arguments <QString> is an alphanumeric character string used to annotate a user mask label.

Examples MASK:USER:LABEL "mikesdata" assigns “mikesdata” as the user mask label.

MASK:USER:LABEL? might return :MASK:USER:LABel "mikesdata", indicating that the label assigned to a user mask is set to “mikesdata”.

MASK:USER:PATTERNBITS

This command sets or returns the number of bits used for serial trigger for the User mask standard. For example, DS1 requires six bits, four leading zeros, a one, and a trailing zero. The query form of this command returns the serial bit value of the displayed mask. The set form of this command affects only the User mask, regardless of the current (displayed) mask.

Group Mask

Syntax
```
MASK:USER:PATTERNBITS <NR1>
MASK:USER:PATTERNBITS?
```

Related Commands MASK:USER:PRESAMPBITS

Arguments <NR1> is an integer that sets the number of bits.

Examples MASK:USER:PATTERNBITS 6 sets the serial bit value of the displayed mask to 6.

MASK:USER:PATTERNBITS? might return :MASK:USER:PATTERNBITS 1, indicating that the serial bit value of the displayed mask is set to 1.

MASK:USER:PRESAMPBITS

This command sets or returns the number of bits before the (isolated one) pulse leading edge in the serial trigger pass/fail testing. For example, DS1 has four leading zeros. The query form of this command returns the presample bit value of the displayed mask. The set form of this command only affects the user mask, regardless of the current (displayed) mask.

Group Mask

Syntax

```
MASK:USER:PRESAMPBITS <NR1>
MASK:USER:PRESAMPBITS?
```

Related Commands MASK:USER:PATTERNBITS

Arguments <NR1> is an integer that sets the number of bits before the trigger pulse.

Examples MASK:USER:PRESAMPBITS 4 sets the presample bit value of the displayed mask to 4.

MASK:USER:PRESAMPBITS? might return :MASK:USER:PRESAMPBITS 0, indicating that the presample bit value of the displayed mask is set to 0.

MASK:USER:RECOrdlength

This command sets or returns the nominal record length for pulse mask testing. The query form of this command returns the record length value of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group Mask

Syntax

```
MASK:USER:RECOrdlength <NR1>
MASK:USER:RECOrdlength?
```

Arguments <NR1> is an integer number that sets the record length.

Examples MASK:USER:RECORDLENGTH 500 sets the record length value of the displayed mask to 500 points.

MASK:USER:RECORDLENGTH? might return :MASK:USER:RECORDLENGTH 0, indicating that the record length value of the displayed mask is set to 0.

MASK:USER:SEG<m> (No Query Form)

This command deletes the specified mask segment from the user mask, whether or not the user mask is the current mask.

m is an integer that specifies the user mask segment number to delete from the user mask.

Group Mask

Syntax MASK:USER:SEG<m> DELEte

Arguments DELEte removes the specified segment from the mask.

Examples MASK:USER:SEG3 DELETE deletes mask segment three from the user mask.

MASK:USER:SEG<m>:NR_Pt? (Query Only)

This query-only command returns the number of points that make up the specified user mask segment. Each mask point consists of a pair of X-Y coordinates.

m is an integer that specifies a user mask segment number.

Group Mask

Syntax MASK:USER:SEG<m>:NR_Pt?

Returns <NR1> is 0 to 50.

Examples MASK:USER:SEG2:NR_PT? might return :MASK:USER:SEG2:NR_PT 5.

MASK:USER:SEG<m>:POINTS

This command sets or returns the X-Y user coordinates of all points in the specified user mask segment. The set form defines new points in the user mask,

replacing any existing points in the specified user mask segment. It sets or returns the vertices for a particular segment in the selected mask.

m is an integer that specifies the user mask segment number.

Group Mask

Syntax
```
MASK:USER:SEG<m>:POINTS <NR3>, <NR3> [, <NR3>, <NR3>]
MASK:USER:SEG<m>:POINTS?
```

Arguments <NR3> refers to the coordinates of one of the vertices in the user mask. Each pair of numbers represents the horizontal and vertical coordinates of a mask segment vertex. The pairs must be listed in a counterclockwise order. If the vertical or horizontal scale or position is changed after this command and then the query form of this command is issued, the value returned from the instrument will not be the same. If just one pair is input, it is ignored and the user mask segment is marked as undefined. The default is not points in the user mask segment.

Examples `MASK:USER:SEG2:POINTS -2.3E-9, 44E-3, -2.5E-9, 47E-3, 1.2E-9, 44E-3` defines the points in user mask segment 2.

`MASK:USER:SEG2:POINTS?` might return `:MASK:USER:SEG2:POINTS 0.0000 0.0000`, indicating that the X-Y user coordinates for user mask segment 2 in the selected mask is set to 0, 0.

MASK:USER:TRIGTOSAMP

This command sets or returns the time in seconds, from the (leading edge) trigger position to the pulse bit sampling position. The query form of this command returns the time value of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group Mask

Syntax
```
MASK:USER:TRIGTOSAMP <NR3>
MASK:USER:TRIGTOSAMP?
```

Related Commands MASK:USER:WIDth

Arguments <NR3> is the floating point number that sets the time to the pulse bit sampling position.

Examples

`MASK:USER:TRIGTOSAMP 162E-9` sets the user mask time from the (leading edge) trigger position to the pulse bit sampling position to 162 nanoseconds.

`MASK:USER:TRIGTOSAMP?` might return `:MASK:USER:TRIGTOSAMP 0.0000`, indicating that the time value of the displayed mask is set to 0 nanosecond.

MASK:USER:VOFFSet

This command sets or returns the nominal vertical offset in volts, used to vertically offset the input channels. The query form of this command returns the offset value of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group

Mask

Syntax

```
MASK:USER:VOFFSet <NR3>
MASK:USER:VOFFSet?
```

Related Commands

MASK:USER:VPOS, MASK:USER:VSCAle

Arguments

`<NR3>` is a floating point number that sets the nominal vertical offset in volts.

Examples

`MASK:USER:VOFFSET 0.0` sets the user mask nominal vertical offset to 0.0 volts.

`MASK:USER:VOFFSET?` might return `:MASK:USER:VOFFSET 0.0000`, indicating that the nominal vertical offset value of the displayed mask is set to 0 volts.

MASK:USER:VPOS

This command sets or returns the nominal vertical position, control in divisions, used to vertically position the input channels. The query form of this command returns the vertical position value of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group

Mask

Syntax

```
MASK:USER:VPOS <NR3>
MASK:USER:VPOS?
```

Related Commands

MASK:USER:VOFFSet, MASK:USER:VSCAle

Arguments <NR3> is a floating point number that sets the nominal vertical position control in divisions.

Examples MASK:USER:VPOS -0.8 sets the user mask nominal vertical position to -0.8 divisions.

MASK:USER:VPOS? might return :MASK:USER:VPOS 0.0000, indicating that the nominal vertical value of the displayed mask is set to 0 divisions.

MASK:USER:VSCAle

This command sets or returns the nominal vertical scale in volts per division, used to vertically scale the input channels. The query form of this command returns the vertical scale value of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group Mask

Syntax
```
MASK:USER:VSCAle <NR3>
MASK:USER:VSCAle?
```

Related Commands MASK:USER:VOFFSet, MASK:USER:VPOS

Arguments <NR3> is a floating point number that sets the nominal bit width in seconds.

Examples MASK:USER:VSCALE200E-3 sets the user mask nominal vertical scale to 200 millivolts per division.

MASK:USER:VSCALE? might return :MASK:USER:VSCALE 1.0000E-3, indicating that the nominal vertical scale of the displayed mask is set to 1 millivolt per division.

MASK:USER:WIDth

This command sets or returns the nominal bit width in seconds. The query form of this command returns the bit width value of the displayed mask. The set form of this command affects only the user mask, regardless of the current (displayed) mask.

Group Mask

Syntax

```
MASK:USER:WIDth <NR3>
MASK:USER:WIDth?
```

Related Commands

MASK:USER:TRIGTOSAMP

Arguments

<NR3> is a floating point number that sets the nominal bit width in seconds. This is the time of one bit of data where bit width = 1 / (data rate of the signal).

Examples

MASK:USER:WIDTH 648E-9 sets the user mask nominal bit width to 648 nanoseconds (1.544 Mbits/second).

MASK:USER:WIDTH? might return :MASK:USER:WIDTH 1.0000E-3, indicating that the nominal vertical scale of the displayed mask is set to 1 millivolt per division.

MATHArbflt<x>:FILepath

This command or query sets the file path for a file of filter coefficients. Setting a path will read that file and load the filter for ARBFLT<x>, where x can be 1 to 8.

Group

Math

Syntax

```
MATHArbflt<x>:FILepath <QString>
MATHArbflt<x>:FILepath?
```

Arguments

<QString> specifies the path to the file of filter coefficients.

Examples

MATHARBFLT8:FILEPATH "c:/myfilters/20mhz.flt" reads filter coefficients from the file and loads the filter for ARBFLT8.

MATHARBFLT1:FILEPATH? might return :MATHARBFLT1:FILEPATH "C:\Users\Public\Tektronix\TekScope\Math Arbitrary Filters\LowPass-Norm\lowpass_0.bw.flt", indicating that the path to the file of filter coefficients is set to "C:\Users\Public\Tektronix\TekScope\Math Arbitrary Filters\LowPass-Norm\lowpass_0.bw.flt".

MATHArbflt<x>:READFile (No Query Form)

This command forces a reread of the filter file for each arbitrary filter in use. The <x> specifies the filter and can be 1 to 8.

Group Math

Syntax `MATHArbflt<x>:READFile FORCe`

Arguments `FORCe` forces a reread of the filter file.

Examples `MATHARBFLT8:READFILE FORCE` forces a reread of the filter file at `MATHARBFLT8:FILEPATH` location for each arbitrary filter in use.

MATHVAR? (Query Only)

This command queries all 8 numerical values you can use within math expressions.

Group Math

Syntax `MATHVAR?`

Related Commands MATHVAR:VAR<x>, MATH<x>:DEFine

Returns `<NR3>` are the stored numerical values.

Examples `MATHVAR?` returns the values of all variables stored in locations 1 through 8.

MATHVAR:VAR<x>

This command sets or queries one of 8 different numerical values you can use within math expressions. These values can range from -9.9e37 to 9.9e37; the default values are 0.0. <x> specifies the location, 1 through 8, in which you can store values. Stored math variables can be referenced within math expressions as VAR1, VAR2,...VAR8.

For example, the following command defines MATH1 as the product of Channel 1 and math variable 1: MATH1:DEFINE "CH1 * VAR1".

Group Math

Syntax
```
MATHVAR:VAR<x> <NR3>
MATHVAR:VAR<x>?
```

Related Commands MATHVAR?, MATH<x>:DEFine

Arguments <NR3> specifies the numerical value to be stored in location x <1 through 8>.

Examples MATHVAR:VAR3 -2.43E-5 stores the value -2.43e-5 in the third math variable location.

MATHVAR:VAR3? might return :MATHVAR:VAR3 24.3000E-6 for the expression stored in location 3.

MATH<x>? (Query Only)

This query-only command returns the definition for the math waveform specified by <x>, which ranges from 1 through 4.

Group Math

Syntax MATH<x>?

Related Commands SELect:<wfm>

Examples MATH1? might return the following definition of a math waveform: :MATH1:DEFINE "Ch1-Ch2";UNITSTRING "V"; NUMAVG 2;VERTICAL:SCALE 112.0000E-3; POSITION 1.6400;AUTOSCALE 0;:MATH1:LABEL:NAME "";XPOS 1.0000;YPOS 600.0000E-3;:MATH1:SPECTRAL:MAG DB;PHASE DEGREES;GATEPOS 0.0000;GATEWIDTH 4.0000E-6;REFLEVEL 20.0000;REFLEVELOFFSET 223.6000E-3;SPAN 625.0000E+6;CENTER 312.5000E+6;RESBW 500.0000E+3;WINDOW GAUSSIAN;SUPPRESS -200.0000;UNWRAP 0;LOCK 0.

MATH<x>:DEFine

This command allows you to define new waveforms using mathematical expressions. Sending this command is equivalent to selecting Math Setup from the Math menu, selecting a math waveform (Math 1 through Math 4), and then entering a math expression in the Math<x> box. The query form of this command returns the math definition for the specified math waveform.

You can specify a math expression from waveforms, measurements and scalar sources, functions, operands, and numerical constants. You can define and display up to four math waveforms simultaneously.

Math expressions can be simple, such as Ch1, which specifies that a waveform should show the signal source of Channel 1 with no mathematical computation. Math expressions can also be complex, consisting of 100 plus characters and comprising many sources (including other math waveforms), functions, and operands. As an example, you can enter the expression Log(Ch1+Ch2), which specifies that the signals from channels 1 and 2 are to be algebraically added, and the base 10 log of the sum is to be shown as the final math waveform.

For more information about constructing mathematical expressions, see *Creating and Using Math Waveforms* in the user online help for this instrument.

Group Math

Syntax

```
MATH<x>:DEFine <QString>
MATH<x>:DEFine?
```

Related Commands MATHVAR:VAR<x>

Arguments `<QString>` quoted string argument is the mathematical expression that defines the waveform.

Examples `MATH2:DEFINE "CH1+CH2"` adds the Channel 1 and Channel 2, storing the results in Math 2 waveform.

`MATH1:DEFINE?` might return

`:MATH1:DEFINE "CH2*REF2"` as the expression that defines Math 1 waveform.

MATH<x>:FILTer:MODe

This command or query sets or returns the filter rise time parameter.

Group Math

Syntax

```
MATH<x>:FILTer:MODe {CENTered|SHIFted}
MATH<x>:FILTer:MODe?
```

Arguments `CENTERED` sets the value at any point to the average of that point in the source waveform and N points on either side of that point.

`SHIFTED` sets the value at any point to the average of that point in the source waveform and 2N points before that in the source waveform. This shifts a rising

edge to the right side of the screen. Shifted mode is sometimes called a causal filter since the value at any point is not caused by points after it in time.

Examples

`MATH1:FILTER:MODE SHIFTED` sets the filter mode to shifted.

`MATH1:FILTER:MODE?` might return `:MATH1:FILTER:MODE CENTERED`, indicating that the filter rise time parameter is set to CENTERED mode for Math 1 waveform.

MATH<x>:FILTer:RISetime

This command or query sets or returns the filter rise time parameter.

Group

Math

Syntax

```
MATH<x>:FILTer:RISetime <NR3>
MATH<x>:FILTer:RISetime?
```

Arguments

`<NR3>` sets how the filter affects a signal. The bandwidth of the filter is approximately 0.35 / (filter rise time). For a square wave input, the measurement system rise time of Math(x) = filter (chx) is very close to the filter rise time of Math(x).

Examples

`MATH1:FILTER:RISETIME 1.7e-3` sets the filter rise time to 1.7 milliseconds.

`MATH1:FILTER:RISETIME?` might return

`:MATH1:FILTER:RISETIME 1.0000E-9`, indicating that the filter rise time parameter is set to 1 nanoseconds for Math 1 waveform.

MATH<x>:LABel:NAMe

This command sets or returns the label string, which is used for annotating the math waveform on the screen. The math waveform to which the label is attached is specified by x, which ranges in value from 1 through 4. This command is equivalent to selecting Math Setup from the Math menu and entering a label in the Label box.

Group

Math

Syntax

```
MATH<x>:LABel:NAMe <QString>
MATH<x>:LABel:NAMe?
```

Arguments <QString> specifies the label to annotate the math waveform.

Examples MATH2:LABEL:NAME "PROBE POINT7" assigns “Probe point7” Math 2 waveform.

MATH2:LABEL:NAME? might return :MATH2:LABEL:NAME "Probe point7", indicating that Probe point 7 is the label for Math 2 waveform.

MATH<x>:LABel:XPOS

This command sets or queries the X screen offset at which the label attached to a math waveform is displayed, relative to the left edge of the screen. Channels are specified by x, which ranges from 1 through 4. This command is equivalent to selecting Math Label from the Math menu and entering a value in the X Position box.

Group Math

Syntax
```
MATH<x>:LABel:XPOS <NR1>
MATH<x>:LABel:XPOS?
```

Related Commands MATH<x>:LABel:YPOS

Arguments <NR1> is the location (in divisions) where the label for the selected math waveform is displayed, relative to the left edge of the screen. Arguments should be integers ranging from 0 to 10.

Examples MATH2:LABEL:XPOS 5 moves the waveform label for the Math 2 waveform (on a four-channel instrument) so that it begins 5 divisions to the right of the left edge of the screen.

MATH2:LABEL:XPOS? might return :MATH2:LABEL:XPOS 2.5, indicating that the waveform label for the Math 2 waveform is currently 2.5 divisions to the right of the left edge of the screen.

MATH<x>:LABel:YPOS

This command sets or queries the Y screen offset at which the label attached to a math waveform is displayed, relative to the waveform handle. The Math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Math Label from the Math menu and entering a value in the Y Position box.

Group Math

Syntax
```
MATH<x>:LABel:YPOS <NR1>
MATH<x>:LABel:YPOS?
```

Related Commands MATH<x>:LABel:XPOS

Arguments <NR1> is the location (in divisions) where the label for the selected math waveform is displayed, relative to the waveform handle. Arguments should rang from 10 to –10.

Examples `MATH2:LABEL:YPOS -2.5` moves the waveform label for the Math 2 waveform to 2.5 divisions below the waveform handle.

`MATH2:LABEL:YPOS?` might return `:MATH2:LABEL:YPOS 0`, indicating that the waveform label for the Math 2 waveform is currently located at the waveform handle.

MATH<x>:NUMAVg

This command sets or returns the acquisition number at which the averaging algorithm will begin exponential averaging. Prior to that acquisition number, the algorithm uses stable averaging. This has no effect unless the AVG() function is used in the specified math expression. If so, it affects all AVG() functions in this math expression. The Math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Set Math Averages from the Math menu and then entering an averaging value for the math waveform.

Group Math

Syntax
```
MATH<x>:NUMAVg <NR1>
MATH<x>:NUMAVg?
```

Related Commands ACQuire:NUMAVg

Arguments <NR1> specifies the number of acquisitions over which exponential averaging is performed.

Examples MATH2:NUMAVG 10 averages the Math 2 waveform ten times.

MATH2:NUMAVG? might return :MATH2:NUMAVG 10, indicating that ten Math 2 waveforms are averaged before a single acquisition occurs.

MATH<x>:SPECTral? (Query Only)

This query-only command returns the current spectral setups for the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Spectral Setup from the Math menu and viewing the current spectral setup values.

Group Math

Syntax

```
MATH<x>:SPECTral?
```

Examples MATH1:SPECTRAL? might return the spectral setup values for MATH1 as follows: :MATH1:SPECTRAL:MAG DB; PHASE DEGREES; GATEPOS 0.0000E+00; GATEWIDTH 1.9996E-06; REFLEVEL 4.4587+01; REFLEVELOFFSET2.2360E-01; SPAN 1.2500E+09; CENTER 6.2500E+08; RESBW 1.0002E=06; WINDOW GAUSSIAN; SUPPRESS -3.5000E+01; UNWRAP 0;LOCK 0.

MATH<x>:SPECTral:CENTER

This command specifies or returns the center frequency of the spectral analyzer output data span for the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Spectral Setup from the Math menu and then entering a Center Freq value.

Group Math

Syntax

```
MATH<x>:SPECTral:CENTER <NR3>
MATH<x>:SPECTral:CENTER?
```

Related Commands MATH<x>:SPECTral:SPAN, HORizontal:MODE:SAMPLERate

Arguments <NR3> is the desired frequency of the spectral analyzer output data span in hertz.

Examples

`MATH3:SPECTRAL:CENTER 10.09E6` sets the center frequency to the closest value it can attain to 10.09 MHz.

`MATH2:SPECTRAL:CENTER?` might return `:MATH2:SPECTral:CENTER 10.0900E+06`, indicating that the center frequency is currently set at 10.09 MHz.

MATH<x>:SPECTral:GATEPOS

This command sets or returns the position of the center of the gate, which is used as the data input to the spectral analyzer for the specified math waveform. The math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Spectral Setup from the Math menu and then entering a Gate Pos value.

Group

Math

Syntax

```
MATH<x>:SPECTral:GATEPOS <NR3>
MATH<x>:SPECTral:GATEPOS?
```

Related Commands

MATH<x>:SPECTral:GATEWIDTH

Arguments

`<NR3>` is the gate position. Units are represented in seconds, with respect to trigger position.

Examples

`MATH1:SPECTRAL:GATEPOS -2e8` specifies the position of the edge of the gate used as the data input to the spectral analyzer. The –2e–8 specifies that the center of the gate is 2e–8 seconds before (to the left of) the trigger.

`MATH2:SPECTRAL:GATEPOS?` might return `:MATH2:SPECTRAL:GATEPOS 0.0000E+00`, indicating that the gate position is set to zero (the trigger position).

MATH<x>:SPECTral:GATEWIDTH

This command sets or returns the gate width input in seconds, to the spectral analyzer for the specified math waveform. The math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Spectral Setup from the Math menu and entering a duration value in the Gate Dur box.

Group

Math

Syntax

```
MATH<x>:SPECTral:GATEWIDTH <NR3>
MATH<x>:SPECTral:GATEWIDTH?
```

Related Commands

MATH<x>:SPECTral:GATEPOS

Arguments

<NR3> is the time across the 10-division screen in seconds.

Examples

MATH1:SPECTRAL:GATEWIDTH 1.0E-3 sets the gate width input to the spectral analyzer at 1 ms.

MATH3:SPECTRAL:GATEWIDTH? might return :MATH3:SPECTRAL:GATEWIDTH 1.0000E-03, indicating that the gate width to the spectral analyzer is set to 1 ms.

MATH<x>:SPECTral:LOCk

This command locks menus for two or more math waveforms together as a group. The query form of this command returns an ON (1) or OFF (0), indicating whether spectral locking is turned on. This command is equal to selecting Spectral Setup from the Math menu, choosing the Control tab and then clicking the Time/Track Frequency Domain Controls button associated with the math waveforms that you want to lock.

Table 2-46: Math<x> Lock Combinations

Math1	Math2	Math3	Locked Math Waveforms
Off	Off	Off	None
Off	Off	On	Math3 and Math4 locked
Off	On	Off	Math2 and Math3 locked
Off	On	On	Math2, Math3, and Math4 locked
On	Off	Off	Math1 and Math2 locked
On	Off	On	Math1 and Math2 locked, Math3 and Math4 locked
On	On	Off	Math1, Math2, and Math3 locked
On	On	On	Math1, Math2, Math3, and Math4 locked

NOTE. *Executing* MATH4:SPECTral:LOCk *via the GPIB interface has no effect since there is no Math5 to which it can be locked.*

Group

Math

Syntax

```
MATH<x>:SPECTral:LOCk {ON|OFF|<NR1>}
MATH<x>:SPECTral:LOCk?
```

Arguments

ON turns on the parameter lock for the specified math waveform.

OFF turns off the parameter lock for the specified math waveform.

<NR1> = 0 disables the parameter lock for the specified math waveform; any other value enables the parameter lock.

Examples

MATH1:SPECTRAL:LOCK ON turns on the parameter lock for Math 1, which causes the parameters for Math 1 and Math 2 waveforms to be locked together.

MATH1:SPECTRAL:LOCK? might return :MATH1:SPECTRAL:LOCK 0, indicating that the parameter lock for Math 1 waveform is turned off.

MATH<x>:SPECTral:MAG

This command sets or returns the units of the SpectralMag function in the specified math definition string. The Math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Spectral Mag from the Math menu and then entering the units that you want in the Scale box, or selecting Basic from the Math menu and then clicking the desired Scale button.

Group

Math

Syntax

```
MATH<x>:SPECTral:MAG {LINEAr|DB|DBM}
MATH<x>:SPECTral:MAG?
```

Arguments

LINEAR sets the SpectralMag units to linear.

DB sets the SpectralMag units to decibels.

DBM sets the SpectralMag units to decibels. It also sets the Ref Level Offset to a value that is the equivalent of 1 mW into 50 Ω.

Examples

MATH2:SPECTRAL:MAG DB sets the SpectralMag units for Math 2 waveform to decibels.

MATH2:SPECTRAL:MAG? might return :MATH2:SPECTRAL:MAG DB, indicating that the SpectralMag units for Math 2 waveform are set to decibels.

MATH<x>:SPECTral:PHASE

This command sets or returns the units of a SpectralPhase function in the specified math definition string. The Math waveform is specified by x, which ranges from 1 through 4. This command is equal to selecting Spectral Phase from the Math menu, selecting the Advanced button, selecting the Vert Axis tab, and then clicking the desired Scale button.

Group Math

Syntax

```
MATH<x>:SPECTral:PHASE {DEGrees|RADians|GROUPDelay}
MATH<x>:SPECTral:PHASE?
```

Arguments

`DEGREES` sets the SpectralPhase units to degrees.

`RADIANS` sets the SpectralPhase units to radians.

`GROUPDELAY` sets the SpectralPhase units to groupdelay, which computes the derivative of unwrapped phase spectrum. Units are expressed in seconds.

Examples

`MATH2:SPECTRAL:PHASE DEGREES` sets the SpectralPhase units for Math 2 waveform to degrees.

`MATH2:SPECTRAL:PHASE?` might return `:MATH2:SPECTRAL:PHASE RADIANS`, indicating that the SpectralPhase units for Math 2 waveform are set to radians.

MATH<x>:SPECTral:REFLevel

This command specifies the vertical position of the specified spectral math waveform on the display screen. The numerical value represents the position at the top of the display graticule. The Math waveform is specified by x, which ranges from 1 through 4. This command is equal to selecting Spectral Setup from the Math menu, choosing the Mag tab and then entering a value in the Reference Level box.

Group Math

Syntax

```
MATH<x>:SPECTral:REFLevel <NR3>
MATH<x>:SPECTral:REFLevel?
```

Related Commands MATH<x>:SPECTral:REFLEVELOffset

Arguments

<NR3> is the value that represents the top of the display screen graticule. The range depends on the units and both the MATH<x>:VERTical:SCAle and MATH<x>:VERTical:POSition settings.

Examples

MATH1:SPECTRAL:REFLEVEL -10 sets the top of the display screen to be a reference level of -10 dB. If the vertical scale is LINEAR, then the vertical units will be the same as the input waveform.

MATH1:SPECTRAL:REFLEVEL? might return :MATH1:SPECTRAL:REFLEVEL 2.0000E+01, indicating that the top of the display screen is set to a reference level of 20 dB.

MATH<x>:SPECTral:REFLEVELOffset

This command sets or returns the spectral level offset used for calculating the dB value for the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. Changing the reference level offset causes the spectral waveform to move vertically, with respect to zero dB. This command is equal to selecting Spectral Setup from the Math menu, choosing the Mag tab and then entering a value in the Reference Level Offset box.

Group

Math

Syntax

```
MATH<x>:SPECTral:REFLEVELOffset {DBM|<NR3>}
MATH<x>:SPECTral:REFLEVELOffset?
```

Related Commands

MATH<x>:SPECTral:REFLevel

Arguments

DBM specifies the reference level used for calculation to be equivalent to 1 mW into 50 Ω (Zero dB will occur at this level).

<NR3> specifies the reference level used for calculation of the decibel value when the output units are Log.

Examples

MATH1:SPECTRAL:REFLEVELOFFSET 0.5 sets the reference level for the Log calculation for decibel to be 0.5. dB = A × Log(y/<NR3>) where A is 10 if the input units are watts and A is 20 if the input units are not watts.

MATH1:SPECTRAL:REFLEVELOFFSET DBM sets the decibel reference to be equivalent to 1 mW into 50 Ω. The reference level numerical value will depend on the input units. If the units are volts, the value is set to 0.2236 V; if the units are amperes, the value is set to 40 µA; if the units are watts, the value is set to 0.001 W.

MATH1:SPECTRAL:REFLEVELOFFSET? might return :MATH1:SPECTRAL:REFLEVELOFFSET 2.23360E-01, indicating that the spectral reference level offset is 223.6 mV.

MATH<x>:SPECTral:RESBw

This command sets or returns the resolution bandwidth of the spectral analyzer for the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Spectral Setup from the Math menu and then entering a value in the Res BW box.

Group Math

Syntax

```
MATH<x>:SPECTral:RESBw <NR3>
MATH<x>:SPECTral:RESBw?
```

Related Commands MATH<x>:SPECTral:GATEWIDTH, MATH<x>:SPECTral:CENTER, MATH<x>:SPECTral:SPAN, MATH<x>:SPECTral:WINdow

Arguments <NR3> is the desired resolution bandwidth value. Units are represented in Hertz.

Examples MATH1:SPECTRAL:RESBW 250E3 sets the resolution bandwidth to the attainable value that is close to 250 kHz.

MATH1:SPECTRAL:RESBW? might return :MATH1:SPECTRAL:RESBW 1.0002E+06, indicating the actual resolution bandwidth value obtained from the spectral analyzer.

MATH<x>:SPECTral:SPAN

This command sets the ceiling of the frequency span to a value that is closest to the specified value. The query form of this command returns the current span value for specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. This command is equal to selecting Spectral Setup from the Math menu and then entering a value in the Freq Span box.

Group Math

Syntax

```
MATH<x>:SPECTral:SPAN {<NR3>|FULl}
MATH<x>:SPECTral:SPAN?
```

Related Commands MATH<x>:SPECTral:CENTER

Arguments <NR3> specifies the frequency span of the output data vector from the spectral analyzer.

FULL sets the top of the span to 1/2 the sample rate and sets the center frequency to 1/2 the span.

Examples MATH1:SPECTRAL:SPAN FULL sets the top of the span to 1/2 the sample rate and sets the center frequency to 1/2 the span.

MATH1:SPECTRAL:SPAN 2.56E6 sets the top of the span to a value that is closest to 2.56 MHz.

MATH1:SPECTRAL:SPAN? might return :MATH1:SPECTRAL:SPAN 1.2500E+09, indicating the actual span value obtained by the spectral analyzer.

MATH<x>:SPECTral:SUPPress

This command sets or returns the phase suppression threshold for the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. This command is equal to selecting Spectral Setup from the Math menu, choosing the Phase tab and then entering a value in the Suppression Threshold box. This command affects only Spectral Phase waveforms.

Group Math

Syntax
```
MATH<x>:SPECTral:SUPPress <NR3>
MATH<x>:SPECTral:SUPPress?
```

Related Commands MATH<x>:SPECTral:UNWRap

Arguments <NR3> is the magnitude level that data with magnitude values below this value are displayed as zero phase.

Examples MATH1:SPECTRAL:SUPPRESS -62 specifies that any data with magnitude values less than -62 dB will have their phase output set to zero.

MATH1:SPECTRAL:SUPPRESS? might return :MATH1:SPECTRAL:SUPPRESS -3.5000E+01, indicating that the phase suppression threshold is currently set to -35 dB.

MATH<x>:SPECTral:UNWRap

This command sets or returns whether phase unwrap of the spectral analyzer output data is enabled for the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. This command is equal to selecting Spectral Setup from the Math menu, choosing the Phase tab and then clicking the Unwrap button. This command affects only Spectral Phase waveforms.

Group Math

Syntax

```
MATH<x>:SPECTral:UNWRap {ON|OFF|<NR1>}
MATH<x>:SPECTral:UNWRap?
```

Related Commands MATH<x>:SPECTral:SUPPress

Arguments

`ON` enables phase unwrap.

`OFF` disables phase wrap.

`<NR1>` = 0 disables phase wrap; any other value enables phase wrap.

Examples

`MATH1:SPECTRAL:UNWRAP ON` enables phase wrap of the spectral analyzer output data.

`MATH1:SPECTRAL:UNWRAP?` might return `:MATH1:SPECTRAL:UNWRAP 0`, indicating that the phase unwrap of the spectral analyzer output data is disabled.

MATH<x>:SPECTral:WINdow

This command sets or returns the window function used to multiply the spectral analyzer input data for the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. 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. This command is equal to selecting Spectral Setup from the Math menu, and choosing from the Window Type drop-down list.

Following is a list of arguments that specify the window function used to multiply the spectral analyzer input data. The windows are listed in the order of their ability to resolve frequencies (resolution bandwidth). For additional information about spectral windows, see *Selecting a Spectral Window* in the online help for this instrument.

Group Math

Syntax

```
MATH<x>:SPECTral:WINdow
{RECTANGular|HAMMing|HANNing|KAISERBessel|BLACKMANHarris|
FLATTOP2|GAUSSian|TEKEXPonential}
MATH<x>:SPECTral:WINdow?
```

Related Commands MATH<x>:SPECTral:RESBw

Arguments

`RECTANGular` window function is equivalent to multiplying all gate data by one.

`HAMMing` window function is based on a cosine series.

`HANNing` window function is based on a cosine series.

`KAISERBessel` window function is based on a cosine series.

`BLACKMANHarris` window function is based on a cosine series.

`FLATTOP2` window function is a cosine series window with a flattened frequency response lobe.

`GAUSSian` window function has the best localization characteristics in the joint time/frequency plane.

`TEKEXPonential` window has an exponential nonsymmetrical shape in the time domain and a triangular shape in the frequency domain.

Examples

`MATH2:SPECTRAL:WINDOW HANNING` applies a Hanning window to the spectral analyzer input data.

`MATH2:SPECTRAL:WINDOW?` might return `:MATH2:SPECTRAL:WINDOW TEKEXPONENTIAL`, indicating that the window function used to multiply the spectral analyzer input data is the Tek exponential function.

MATH<x>:THRESHold

This command sets or queries the threshold for the math waveform specified by x, which can be 1 through 4.

Group Math

Syntax

```
MATH<x>:THRESHold <NR3>
MATH<x>:THRESHold?
```

Related Commands D<x>:THRESHold

Arguments <NR3> specifies the math threshold in volts.

Examples MATH1:THRESHOLD 1 sets the threshold value of the MATH1 to 1 volt.

MATH1:THRESHOLD? might return MATH1:THRESHOLD 500.0000E-3, indicating that the threshold value set for the MATH1 is 0.5 volts.

MATH<x>:UNITString

This command sets or queries the string to use for units for the math waveform specified by x, which can be 1 through 4. This command will override the default unit string with the one that you specify.

Group Math

Syntax
```
MATH<x>:UNITString <QString>
MATH<x>:UNITString?
```

Arguments <QString> quoted string argument is the units to be used for the specified math waveform.

Examples MATH2:UNITString "deg" specifies that the unit of the Math 2 waveform operation will be called deg.

MATH1:UNITSTRING? might return :MATH1:UNITSTRING "deg" as the unit for the Math 1 waveform operation.

MATH<x>:VERTical:AUTOSCale

This command sets or queries auto-scaling of the specified math waveform.

Group Math

Syntax
```
MATH<x>:VERTical:AUTOSCale {<NR1>|OFF|ON}
MATH<x>:VERTical:AUTOSCale?
```

Related Commands MATH<x>:VERTical:POSition

MATH<x>:VERTical:SCAle

Arguments

`ON, 1` – enables auto-scaling of new math waveforms. (This is the default).

`OFF, 2` – math waveforms will not be scaled after activation, and will use the current `:MATH<x>:VERTical:SCAle` and `:MATH<x>:VERTical:POSition` values.

Examples

`MATH1:VERTICAL:AUTOSCALE 1` enables auto-scaling of the Math1 waveform.

MATH<x>:VERTical:POSition

This command sets or queries the vertical position of the specified Math waveform. The Math waveform is specified by x, which ranges from 1 through 4. The position value is usually applied to the signal before it is digitized. The highest three units/div scale ranges of a given math are implemented by changing the way the acquired data is displayed. When the instrument is operating in any of these highest three scale ranges, the position control operates only on the signal after it is digitized. Note that if a signal that exceeds the range of the digitizer in one of these three scale ranges is repositioned, the displayed waveform will contain clipped values on-screen. This command is equivalent to selecting Position/Scale from the Math menu and then entering a Vert Pos value or adjusting the front panel Vertical POSITION knob.

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, whether a channel, math, or reference waveform. The position value determines the vertical graticule coordinate at which input signal values, equal to the present offset setting for that reference, are displayed. For example, if the position for Math 3 is set to 2.0 and the offset is set to 3.0, then the input signals equal to 3.0 are displayed 2.0 divisions above the center of the screen.

Be aware that autoscaling occurs when a math waveform is first defined and enabled, or when a math string changes. After the math waveform is computed for the first time, the instrument determines the min + max of that waveform data. Then, the instrument sets the math position so that (min + max)/2 is in the center of the screen. In addition, the instrument sets the math scale so that the range of the min and max cover 6 divisions. This autoscaling process can take up to 1/2 second to complete and will override any vertical scale or position commands for that math waveform received during this time. You should insert an appropriate pause in your program after defining and enabling a math waveform before changing its position or scale.

Group

Math

Syntax

```
MATH<x>:VERTical:POSition <NR3>
MATH<x>:VERTical:POSition?
```

Related Commands CH<x>:POSition, REF<x>:VERTical:POSition

Arguments <NR3> is the desired position control in divisions from the center graticule.

Examples MATH2:VERTICAL:POSITION 1.3E+00 positions the Math 2 waveform 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 waveform is 1.3 divisions below the center graticule.

MATH<x>:VERTical:SCAle

This command sets or queries the vertical scale of the specified math waveform. The Math waveform is specified by x, which ranges from 1 through 4. This command is equivalent to selecting Position/Scale from the Math menu and then entering a Vert Scale value or adjusting the front panel Vertical SCALE knob.

Each waveform has its own 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. For reference and math waveforms, the scale setting controls the display only, graphically scaling these waveforms and having no affect on the acquisition hardware.

Be aware that autoscaling occurs when a math waveform is first defined and enabled, or when a math string changes. After the math waveform is computed for the first time, the instrument determines the min + max of that waveform data. Then, the instrument sets the math position so that (min + max)/2 is in the center of the screen. In addition, the instrument sets the math scale so that the range of the min and max covers 6 divisions. This autoscaling process can take up to 1/2 second to complete and will override any vertical scale or position commands for that math waveform received during this time. You should insert an appropriate pause in your program after defining and enabling a math waveform before changing its position or scale.

Group Math

Syntax
```
MATH<x>:VERTical:SCAle <NR3>
MATH<x>:VERTical:SCAle?
```

Related Commands CH<x>:SCAle, REF<x>:VERTical:SCAle

Arguments <NR3> is the scale in volts, amps or watts per division. The range is from 100.0E-36 through 100.0E+36.

Examples MATH4:VERTICAL:SCALE 100E-03 sets the Math 4 waveform scale to 100 mV per division.

MATH2:VERTICAL:SCALE? might return :MATH2:VERTICAL:SCALE 1.0000E+00, indicating that the current scale setting of Math 2 waveform is 1 V per division.

MEASUrement? (Query Only)

This query-only command returns all measurement parameters in the following order: MEAS1, MEAS2, MEAS3, MEAS4, MEAS5, MEAS6, MEAS7, MEAS8, and IMMED.

Group Measurement

Syntax MEASUrement?

Examples MEASUREMENT? might return :MEASUREMENT:GATING OFF;IMMED:TYPE UNDEFINED;UNITS "V";SOURCE1 CH1;SOURCE2 CH1;SOURCE1:SIGTYPE PULSE;:MEASUREMENT:IMMED:SOURCE2:SIGTYPE PULSE;:MEASUREMENT:IMMED:DELAY:EDGE1 RISE;EDGE2 RISE;DIRECTION FORWARDS;:MEASUREMENT:IMMED:REFLEVEL:METHOD PERCENT;ABSOLUTE:HIGH 0.0000;LOW 0.0000;MID1 0.0000;MID2 0.0000;:MEASUREMENT:IMMED:REFLEVEL:PERCENT:HIGH 90.0000;LOW 10.0000;MID1 50.0000;MID2 50.0000;:MEASUREMENT:IMMED:METHOD HISTOGRAM;NOISE HIGH;:MEASUREMENT:MEAS1:STATE 0;TYPE UNDEFINED;UNITS "V";SOURCE1 CH1;SOURCE2 CH1;SOURCE1:SIGTYPE PULSE;:MEASUREMENT:MEAS1:SOURCE2:SIGTYPE PULSE;:MEASUREMENT:MEAS1:DELAY:EDGE1 RISE;EDGE2 RISE;DIRECTION FORWARDS;:MEASUREMENT:MEAS1:REFLEVEL:METHOD PERCENT;ABSOLUTE:HIGH 0.0000;LOW 0.0000;MID1 0.0000;MID2 0.0000;:MEASUREMENT:MEAS1:REFLEVEL:PERCENT:HIGH 90.0000;LOW 10.0000;MID1 50.0000;MID2 50.0000;:MEASUREMENT:MEAS1:METHOD HISTOGRAM;NOISE HIGH;:MEASUREMENT:MEAS2:STATE 0;TYPE UNDEFINED;UNITS "V";SOURCE1 CH1;SOURCE2 CH1;SOURCE1:SIGTYPE PULSE;:MEASUREMENT:MEAS2:SOURCE2:SIGTYPE PULSE;:MEASUREMENT:MEAS2:DELAY:EDGE1 RISE;EDGE2 RISE;DIRECTION.

MEASUrement:ANNOTation:IMMEDSTAte

This command sets or queries the state of immediate measurement annotation. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax

```
MEASUrement:ANNOTation:IMMEDSTAte {<NR1>|OFF|ON}
MEASUrement:ANNOTation:IMMEDSTAte?
```

Arguments

NR1 = 0 disables immediate state measurement annotation, any other value enables immediate state measurements.

OFF disables annotation on immediate state measurements.

ON enables annotation on immediate state measurements.

Examples

MEASUREMENT:ANNOTATION:IMMEDSTATE ON enables annotation of immediate measurements.

MEASUREMENT:ANNOTATION:IMMEDSTATE? might return :MEASUREMENT:ANNOTATION:IMMEDSTATE 0, indicating that annotation of immediate measurements is off.

MEASUrement:ANNOTation:NUMX? (Query Only)

This query-only command returns the number of horizontal measurement annotations currently being displayed. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax

```
MEASUrement:ANNOTation:NUMX?
```

Examples

MEASUREMENT:ANNOTATION:NUMX? might return MEASUREMENT:ANNOTATION:NUMX 2, indicating that there are currently 2 horizontal lines drawn on the graticule showing where the measurement specified by MEASUREMENT:ANNOTATION:STATE is being performed.

MEASUrement:ANNOTation:NUMY? (Query Only)

This query-only command returns the number of vertical measurement annotations currently being displayed. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax `MEASUrement:ANNOTation:NUMY?`

Examples `MEASUREMENT:ANNOTATION:NUMY?` might return `MEASUREMENT:ANNOTATION:NUMY 2`, indicating that there are currently 2 vertical lines drawn on the graticule showing where the measurement specified by `MEASUREMENT:ANNOTATION:STATE` is being performed.

MEASUrement:ANNOTation:STATE

This command sets or returns the state of visible measurement annotations.

Group Measurement

Syntax
```
MEASUrement:ANNOTation:STATE
{OFF|MEAS1|MEAS2|MEAS3|MEAS4|MEAS5|MEAS6|MEAS7|MEAS8}
MEASUrement:ANNOTation:STATE?
```

Arguments `OFF` turns off visible measurement annotations.

`MEAS<x>` turns on the display of visible measurement annotations for measurement <x>, where <x> can be 1, 2, 3, 4, 5, 6, 7, or 8. There must be an active measurement before you can activate an annotation for a specified measurement.

Examples `MEASUREMENT:ANNOTATION:STATE MEAS2` turns on the display of visible measurement annotations for measurement 2.

`MEASUREMENT:ANNOTATION:STATE?` might return `:MEASUREMENT:ANNOTATION:STATE OFF`, indicating that no measurement annotations are active.

MEASUrement:ANNOTation:TYPe

This command sets or queries the type of annotations being used. This command is similar to setting the annotation type in the Measure > Annotation menu.

Group Measurement

Syntax
```
MEASUrement:ANNOTation:TYPe {STANdard|DETAiled}
MEASUrement:ANNOTation:TYPe?
```

Arguments `STANdard` sets the annotation type to STANDARD.

`DETAiled` sets the annotation type to DETAiled, so that more detailed annotations are displayed.

Examples `MEASUREMENT:ANNOTATION:TYPE DETAILED` sets the measurement annotation type to detailed.

`MEASUREMENT:ANNOTATION:TYPE?` might return `:MEASUREMENT:ANNOTATION:TYPE STANDARD`, indicating that standard annotations will be displayed.

MEASUrement:ANNOTation:X<x>? (Query Only)

This query-only command returns the value of the specified X annotation. X annotations are numbered, in general, from top to bottom on the display. The value of <x> can vary from 1 to 6. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax
```
MEASUrement:ANNOTation:X<x>?
```

Examples `MEASUREMENT:ANNOTATION:X1?` might return `:MEASUREMENT:ANNOTATION:X1 277.2250E-12`, indicating that annotation X1 is 277 ps to the right of the trigger.

MEASUrement:ANNOTation:Y<x>? (Query Only)

This query-only command returns the value of the specified Y annotation. Y annotations are numbered, in general, from left to right on the display. The value

of <x> can vary from 1 to 6. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax `MEASUrement:ANNOTation:Y<x>?`

Examples `MEASUREMENT:ANNOTATION:Y1?` might return `:MEASUREMENT:ANNOTATION:X1 305.6406E-3,`, indicating that annotation Y1 is 305 mV above the channel reference.

MEASUrement:GATing

This command specifies or returns the measurement gating setting. This command is equivalent to selecting Gating from the Measure menu and then clicking the desired Measurement Gating setting.

Group Measurement

Syntax
```
MEASUrement:GATing {ON|OFF|<NR1>|ZOOM<x>|CURSor}
MEASUrement:GATing?
```

Arguments `ON` turns on measurement gating.

`OFF` turns off measurement gating.

`<NR1>` = 0 turns off measurement gating; any other value turns on measurement gating.

`ZOOM<x>` turns on gating, using the left and right edges of the zoom box. <x> specifies the zoom window, which ranges from 1 through 4.

`CURSOR` limits measurements to the portion of the waveform between the vertical bar cursors, even if they are off screen.

Examples `MEASUREMENT:GATING ON` turns on measurement gating.

`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)

This query-only command returns all immediate measurement setup parameters. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax `MEASUrement:IMMed?`

Examples `MEASUREMENT:IMMED?` might return `:MEASUREMENT:IMMED:TYPE AMPLITUDE; UNITS "V";SOURCE1 CH1;SOURCE2 CH1; SOURCE1:SIGTYPE PULSE; :MEASUREMENT:IMMED:SOURCE2:SIGTYPE PULSE; :MEASUREMENT:IMMED:DELAY:EDGE1 RISE; EDGE2 RISE;DIRECTION FORWARDS; :MEASUREMENT:IMMED:REFLEVEL:METHOD PERCENT; ABSOLUTE:HIGH 0.0000;LOW 0.0000;MID1 0.0000;MID2 0.0000;:MEASUREMENT:IMMED:REFLEVEL:PERCENT:HIGH 90.0000;LOW 10.0000;MID1 50.0000;MID2 50.0000;:MEASUREMENT:IMMED:METHOD HISTOGRAM;NOISE HIGH`.

MEASUrement:IMMed:DELay? (Query Only)

This query-only command returns information about the immediate delay measurement. This command is equivalent to selecting Measurement Setup from the Measure menu, choosing the Time tab and then clicking the Delay button. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax `MEASUrement:IMMed:DELay?`

Examples `MEASUREMENT:IMMED:DELAY?` might return `:MEASUREMENT:IMMED:DELAY:EDGE1 RISE;EDGE2 RISE; DIRECTION FORWARDS`.

MEASUrement:IMMed:DELay:DIREction

This command sets or returns the starting point and direction that determines the delay "to" edge when taking an immediate delay measurement. Use the MEASUrement:IMMed:SOUrce<x> command to specify the delay "to" waveform. This command is equivalent to selecting Measurement Setup from the

Measure menu, choosing the Time tab, clicking the Delay button to display the delay settings and then clicking the desired Search Direction setting.

Group Measurement

Syntax
```
MEASUrement:IMMed:DELay:DIRECtion {BACKWards|FORWards}
MEASUrement:IMMed:DELay:DIRECtion?
```

Related Commands MEASUrement:IMMed:SOUrce<x>

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[1]

This command sets or queries the slope of the edge that is used for the delay "from" waveform when taking an immediate delay measurement. Use the MEASUrement:IMMed:SOUrce<x> command to specify the waveform. This command is equivalent to selecting Measurement Setup from the Measure menu, choosing the Time tab, clicking the Delay button to display the delay settings and then clicking the desired Delay Edge1 setting.

NOTE. *If you do not specify a numerical suffix, the source is assumed to be SOURCE 1.*

Group Measurement

Syntax
```
MEASUrement:IMMed:DELay:EDGE[1] {FALL|RISe}
MEASUrement:IMMed:DELay:EDGE[1]?
```

Related Commands MEASUrement:IMMed:SOUrce<x>

Arguments `FALL` specifies the falling edge.

`RISe` specifies the rising edge.

Examples `MEASUREMENT:IMMED:DELAY:EDGE[1] 1 RISE` specifies that the rising edge be used for the immediate delay measurement.

`MEASUREMENT:IMMED:DELAY:EDGE[1]?` might return `:MEASUREMENT:IMMED:DELAY:EDGE1 FALL`, indicating that the falling edge or negative edge of the waveform is used for the immediate delay measurement.

MEASUrement:IMMed:DELay:EDGE2

This command sets or queries the slope of the edge that is used for the delay "to" waveform when taking an immediate delay measurement. Use the MEASUrement:IMMed:SOUrce<x> command to specify the waveform. This command is equivalent to selecting Measurement Setup from the Measure menu, choosing the Time tab, clicking the Delay button to display the delay settings and then clicking the desired Delay Edge2 setting.

Group Measurement

Syntax
```
MEASUrement:IMMed:DELay:EDGE2 {FALL|RISe}
MEASUrement:IMMed:DELay:EDGE2?
```

Related Commands MEASUrement:IMMed:SOUrce<x>

Arguments `FALL` specifies the falling edge.

`RISe` specifies the rising edge.

Examples `MEASUREMENT:IMMED:DELAY:EDGE2 RISE` specifies that the rising edge be used for the immediate delay measurement.

`MEASUREMENT:IMMED:DELAY:EDGE2?` might return `:MEASUREMENT:IMMED:DELAY:EDGE2 FALL`, indicating that the falling edge or negative edge of the waveform is used for the immediate delay measurement.

MEASUrement:IMMed:METHod

This command specifies or queries the method used to calculate the 0% and 100% reference level for immediate measurements.

Group Measurement

Syntax

```
MEASUrement:IMMed:METHod {HIStogram|MINMax|MEAN}
MEASUrement:IMMed:METHod?
```

Arguments

`HIStogram` sets the high and low waveform levels statistically using a histogram algorithm.

`MINMax` sets the high and low waveform levels to MAX and MIN, respectively.

`MEAN` sets the high and low waveform levels to their mean.

Examples

`MEASUREMENT:IMMED:METHOD HISTOGRAM` specifies that the high and low reference levels are set statistically.

`MEASUREMENT:IMMED:METHOD?` might return `:MEASUREMENT:IMMED:METHOD MINMAX`, indicating that the reference levels are set to MIN and MAX for immediate measurements.

MEASUrement:IMMed:NOISe

This command sets or queries whether the noise measurement is made on the high or low level of the waveform. Sending this command is equivalent to selecting Ref Levs > Eye > Top Level or Base Level in the Comm tab of the Measurement Setup dialog box. The Eye section is displayed only if you have an eye-pattern or optical measurement defined.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Measurement

Syntax

```
MEASUrement:IMMed:NOISe {HIGH|LOW}
MEASUrement:IMMed:NOISe?
```

Arguments

`HIGH` argument causes the measurement for noise to be taken at the high level of the waveform.

LOW argument causes the measurement for noise to be taken at the low level of the waveform.

Examples

MEASUREMENT:IMMED:NOISE HIGH causes the noise measurement to be made at the high level of the waveform.

MEASUREMENT:IMMED:NOISE? might return :MEASUREMENT:IMMED:NOISE LOW, indicating that the noise measurement will be made at the low level of the waveform.

MEASUrement:IMMed:REFLevel? (Query Only)

This query-only command returns the reference level settings for the immediate measurement. It returns them in the following order: ABSOLUTE and then PERCENT for individual user measurements.

Group

Measurement

Syntax

```
MEASUrement:IMMed:REFLevel?
```

Examples

MEASUREMENT:IMMED:REFLEVEL? might return :MEASUREMENT:IMMED:REFLEVEL:PERCENT:HIGH 90 indicating that the reference level is set to 90% of HIGH.

MEASUrement:IMMed:REFLevel:ABSolute:HIGH

This command sets or queries the high reference level, and is the upper reference level when MEASUrement:IMMed:REFLevel:METHod is set to Absolute. Note that this command affects the results of rise and fall measurements.

NOTE. *This command affects the results of period, frequency, delay, and all cyclic measurements.*

Group

Measurement

Syntax

```
MEASUrement:IMMed:REFLevel:ABSolute:HIGH <NR3>
MEASUrement:IMMed:REFLevel:ABSolute:HIGH?
```

Related Commands

MEASUrement:IMMed:REFLevel:METHod

Arguments <NR3> is the high reference level in volts. The default is 0.0 V.

Examples MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:HIGH 1.71 sets the high reference level to 1.71 V.

MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:HIGH? might return :MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:HIGH 1.7100E+00, indicating that the absolute high reference level is set to 1.71 V.

MEASUrement:IMMed:REFLevel:ABSolute:LOW

This command sets or queries the low reference level, and is the zero percent level when MEASUrement:IMMed:REFLevel:METHod is set to Absolute. Note that this command affects the results of rise and fall measurements.

NOTE. *This command affects the results of period, frequency, delay, and all cyclic measurements.*

Group Measurement

Syntax
```
MEASUrement:IMMed:REFLevel:ABSolute:LOW <NR3>
MEASUrement:IMMed:REFLevel:ABSolute:LOW?
```

Related Commands MEASUrement:IMMed:REFLevel:METHod

Arguments <NR3> is the low reference level in volts. The default is 0.0 V.

Examples MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:LOW 0.0 sets the low reference level to 0.0 V.

MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:LOW? might return :MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:LOW 0.0000E+00, indicating that the absolute low reference level is set to 0.0 V.

MEASUrement:IMMed:REFLevel:ABSolute:MID<x>

This command sets or queries the mid reference level, and is the 50% reference level when MEASUrement:IMMed:REFLevel:METHod is set to Absolute. Note that this command affects the results of period, frequency, delay, and all cyclic measurements. Note that this command affects the results of delay measurements.

NOTE. *This command affects the results of period, frequency, delay, and all cyclic measurements.*

Group Measurement

Syntax
```
MEASUrement:IMMed:REFLevel:ABSolute:MID<x> <NR3>
MEASUrement:IMMed:REFLevel:ABSolute:MID<x>?
```

Related Commands MEASUrement:IMMed:REFLevel:METHod

Arguments `<NR3>` is the mid1 (the "from" waveform when taking a delay measurement) or mid2 (the "to" waveform when taking a delay measurement) reference level in volts. The default is 0.0 V.

Examples `MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:MID2 0.5` sets the mid reference level for the delay waveform to 0.5 V.

`MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:MID2?` might return `:MEASUREMENT:IMMED:REFLEVEL:ABSOLUTE:MID2 0.5000E+00`, indicating that the absolute mid2 reference level is set to 0.5 V.

MEASUrement:IMMed:REFLevel:METHod

This command specifies or queries the reference level units used for measurement calculations. This command is equivalent to selecting Reference Levels from the Measure menu and then choosing the desired reference level from the Units group box.

Group Measurement

Syntax
```
MEASUrement:IMMed:REFLevel:METHod {ABSolute|PERCent}
MEASUrement:IMMed:REFLevel:METHod?
```

Arguments `ABSolute` specifies that the reference levels are set explicitly using the `MEASUrement:IMMed:REFLevel:ABSolute` commands. This method is useful when precise values are required. For instance, 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:IMMed:REFLevel:PERCent` commands.

Examples

`MEASUREMENT:IMMED:REFLEVEL:METHOD ABSOLUTE` specifies that explicit user-defined values are used for the reference levels.

`MEASUREMENT:IMMED:REFLEVEL:METHOD?` might return `:MEASUREMENT:IMMED:REFLEVEL:METHOD PERCENT`, indicating that the reference level units used are calculated as a percent relative to HIGH and LOW.

MEASUrement:IMMed:REFLevel:PERCent:HIGH

This command sets or queries the percentage (where 100% is equal to HIGH) used to calculate the high reference level when MEASUrement:IMMed:REFLevel:METHod is set to Percent. Note that this command affects the results of rise and fall measurements.

NOTE. *This command affects the results of period, frequency, delay, and all cyclic measurements.*

Group

Measurement

Syntax

```
MEASUrement:IMMed:REFLevel:PERCent:HIGH <NR1>
MEASUrement:IMMed:REFLevel:PERCent:HIGH?
```

Related Commands

MEASUrement:IMMed:REFLevel:METHod

Arguments

`<NR1>` is the high reference level, ranging from 0 to 100%. The default high reference level is 90%.

Examples

`MEASUREMENT:IMMED:REFLEVEL:PERCENT:HIGH 95` sets the high reference level to 95% of HIGH.

`MEASUREMENT:IMMED:REFLEVEL:PERCENT:HIGH?` might return `:MEASUREMENT:IMMED:REFLEVEL:PERCENT:HIGH 90`, indicating that the percentage high reference level is set to 90% of HIGH.

MEASUrement:IMMed:REFLevel:PERCent:LOW

This command sets or queries the percentage (where 100% is equal to HIGH) used to calculate the low reference level when MEASUrement:IMMed:REFLevel:METHod is set to Percent. Note that this command affects the results of rise and fall measurements. This command is equivalent to selecting Reference Levels from the Measure menu and then entering the Percentage Low Ref value.

NOTE. *This command affects the results of period, frequency, delay, and all cyclic measurements.*

Group Measurement

Syntax
```
MEASUrement:IMMed:REFLevel:PERCent:LOW <NR1>
MEASUrement:IMMed:REFLevel:PERCent:LOW?
```

Related Commands MEASUrement:IMMed:REFLevel:METHod

Arguments <NR1> is the low reference level, ranging from 0 to 100%. The default low reference level is 10%.

Examples MEASUREMENT:IMMED:REFLEVEL:PERCENT:LOW 15 sets the low reference level to 15% of HIGH.

MEASUREMENT:IMMED:REFLEVEL:PERCENT:LOW? might return :MEASUREMENT:IMMED:REFLEVEL:PERCENT:LOW 10, indicating that the percentage low reference level is set to 10% of HIGH.

MEASUrement:IMMed:REFLevel:PERCent:MID<x>

This command sets or queries the percentage (where 100% is equal to HIGH) used to calculate the mid reference level when MEASUrement:IMMed:REFLevel:METHod is set to Percent. Mid1 is for the first waveform specified, and Mid2 is for the second waveform specified. Note that this command affects the results of delay measurements.

NOTE. *This command affects the results of period, frequency, delay, and all cyclic measurements.*

Group Measurement

Syntax
```
MEASUrement:IMMed:REFLevel:PERCent:MID<x> <NR1>
MEASUrement:IMMed:REFLevel:PERCent:MID<x>?
```

Related Commands MEASUrement:IMMed:REFLevel:METHod

Arguments <NR1> is the mid reference level, ranging from 0 to 100%. The default mid reference level is 50%.

Examples MEASUREMENT:IMMED:REFLEVEL:PERCENT:MID2 40 sets the mid2 reference level to 40% of HIGH.

MEASUREMENT:IMMED:REFLEVEL:PERCENT:MID2? might return :MEASUREMENT:IMMED:REFLEVEL:PERCENT:MID2 45, indicating that the percentage mid2 reference level is set to 45% of HIGH.

MEASUrement:IMMed:SOUrce<x>

This command sets or queries the source for phase or delay immediate measurements. This command is equivalent to selecting Measurement Setup from the Measure menu, choosing the Time tab, clicking the Delay button to display the delay settings and then clicking the desired Source1 (From) setting or Source2 (To) setting.

Tip: 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:SOUrce<x> {CH<x>|MATH<y>|REF<x>|HIStogram}
MEASUrement:IMMed:SOUrce<x>?
```

Arguments CH<x> is an input channel waveform. The x variable can be expressed as an integer ranging from 1 through 4.

MATH<y> is a math waveform. The y variable can be expressed as an integer ranging from 1 through 4.

REF<X> is a reference waveform. The x variable can be expressed as an integer ranging from 1 through 4.

HIStogram indicates histogram as the object to be measured. HIStogram not allowed on SOUrce2.

Examples MEASUREMENT:IMMED:SOURCE2 REF3 sets the waveform in reference memory location 3 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>:SIGType

This command sets or queries the type of input signal used for measurement SOURCE<x>, 1 or 2.

To ensure accurate measurements, use this command to specify the input-signal type for the measurement source.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Measurement

Syntax

```
MEASUrement:IMMed:SOUrce<x>:SIGType {PULse|EYE}
MEASUrement:IMMed:SOUrce<x>:SIGType?
```

Arguments

`PULSE` is for generic signals that are not associated with synchronous communications standards.

`EYE` is for synchronous-communication signals with NRZ-like characteristics (nonreturn to zero).

Examples

`MEASUREMENT:IMMED:SOURCE1:SIGTYPE PULSE` selects Pulse as the signal type for source 1.

`MEASUREMENT:IMMED:SOURCE1:SIGTYPE?` might return `:MEASUREMENT:IMMED:SOURCE1:SIGTYPE EYE`, indicating that the signal type for source 1 is EYE.

MEASUrement:IMMed:TYPe

This command sets or queries the immediate measurement type. Immediate measurements and annotations are not displayed on the screen.

Group Measurement

Syntax

```
MEASUrement:IMMed:TYPe {AMPlitude|AREa|
BURst|CAREa|CMEan|CRMs|DELay|DISTDUty|
EXTINCTDB|EXTINCTPCT|EXTINCTRATIO|
EYEHeight|EYEWIdth|FALL|FREQuency|HIGH|
HITs|LOW|MAXimum|MEAN|MEDian|MINImum|
NCROss|NDUty|NOVershoot|NWIdth|PBASe|
PCROss|PCTCROss|PDUty|PEAKHits|PERIod|
PHASe|PK2Pk|PKPKJitter|PKPKNoise|
POVershoot|PTOP|PWIdth|QFACtor|RISe|
```

```
RMS|RMSJitter|RMSNoise|SIGMA1|SIGMA2|
SIGMA3|SIXSigmajit|SNRatio|STDdev| UNDEFINED|WAVEFORMS}
MEASUrement:IMMed:TYPe?
```

Arguments

AMPlitude 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.

Amplitude = High - 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.

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.

CMEan (cycle mean) measures the arithmetic mean over the first cycle in the waveform or the first cycle in the gated region.

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.

DELay measures the time between the middle reference (default = 50%) amplitude point of the source waveform and the destination waveform.

DISTDUty (duty cycle distortion) measures the time between the falling edge and the rising edge of the eye pattern at the mid reference level. It is the peak-to-peak time variation of the first eye crossing measured at the mid-reference as a percent of the eye period.

EXTINCTDB measures the extinction ratio of an optical waveform (eye diagram). Extinction Ratio (dB) measures the ratio of the average power levels for the logic High to the logic Low of an optical waveform and expresses the result in dB. This measurement only works for fast acquisition signals or a reference waveform saved in fast acquisition mode.

Extinction dB = 10 × (log 10 (High / Low)

EXTINCTPCT measures the extinction ratio of the selected optical waveform. Extinction Ratio (%) measures the ratio of the average power levels for the logic Low (off) to the logic (High) (on) of an optical waveform and expresses the result in percent. This measurement only works for fast acquisition signals or a reference waveform saved in fast acquisition mode.

Extinction % = 100.0 × (Low / High)

EXTINCTRATIO measures the extinction ratio of the selected optical waveform. Extinction Ratio measures the ratio of the average power levels for the logic High to the logic Low of an optical waveform and expresses the result without units. This measurement only works for fast acquisition signals or a reference waveform saved in fast acquisition mode. Extinction ratios greater than 100 or less than 1 generate errors; low must be greater than or equal to 1 µW.

Extinction Ratio = (High / Low)

EYEHeight measures the vertical opening of an eye diagram in volts.

EYEWidth measures the width of an eye diagram in seconds.

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%).

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.

You can also limit the High measurement (normally taken over the entire waveform record) to a gated region on the waveform.

HITS (histogram hits) measures the number of points in or on the histogram box.

LOW measures the Low reference (0% level, sometimes called Baseline) of a waveform.

MAXimum finds the maximum amplitude. This value is the most positive peak voltage found. It is measured over the entire waveform or gated region. When histogram is selected with the MEASUrement:METHod command, the maximum measurement measures the voltage of the highest nonzero bin in vertical histograms or the time of the right-most bin in horizontal histograms.

MEAN amplitude measurement finds the arithmetic mean over the entire waveform or gated region. When histogram is selected with the MEASUrement:METHod command, the mean measurement measures the average of all acquired points within or on the histogram.

MEDian (histogram measurement) measures the middle point of the histogram box. Half of all acquired points within or on the histogram box are less than this value and half are greater than this value.

MINImum finds the minimum amplitude. This value is typically the most negative peak voltage. It is measured over the entire waveform or gated region. When histogram is selected with the MEASUrement:METHod command, the minimum measurement measures the lowest nonzero bin in vertical histograms or the time of the left-most nonzero bin in the horizontal histograms.

NCROss (timing measurement) measures the time from the trigger point to the first falling edge of the waveform or gated region. The distance (time) is measured at the middle reference amplitude point of the signal.

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.

Negative Duty Cycle = (Negative Width) / Period × 100%

NOVershoot (negative overshoot) finds the negative overshoot value over the entire waveform or gated region.

Negative Overshoot = (Low - Minimum) / Amplitude × 100%)

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.

PBASe measures the base value used in extinction ratio measurements.

PCROss (timing measurement) measures the time from the trigger point to the first positive edge of the waveform or gated region. The distance (time) is measured at the middle reference amplitude point of the signal.

PCTCROss measures the location of the eye crossing point expressed as a percentage of EYEHeight.

Crossing percent = 100 ×[(eye-crossing-point - PBASe)/(PTOP - PBASe)]

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.

Positive Duty Cycle = (Positive Width)/Period × 100%

PEAKHits measures the number of points in the largest bin of the histogram.

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. When histogram is selected with the MEASUrement:METHod command, the PK2Pk measurement measures the histogram peak to peak difference.

PKPKJitter measures the variance (minimum and maximum values) in the time locations of the cross point.

PKPKNoise measures the peak-to-peak noise on a waveform at the mid reference level.

POVershoot

The positive overshoot amplitude measurement finds the positive overshoot value over the entire waveform or gated region.

Positive Overshoot = (Maximum - High) / Amplitude ×100%

PTOT measures the top value used in extinction ratio measurements.

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.

QFACtor measures the quality factor. The Q factor is a figure of merit for an eye diagram, which indicates the vertical eye opening relative to the noise at the low and high logic levels. It is the ratio of the eye size to noise.

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%).

RMS amplitude measurement finds the true Root Mean Square voltage in the entire waveform or gated region.

RMSJitter measures the variance in the time locations of the cross point. The RMS jitter is defined as one standard deviation at the cross point.

RMSNoise measures the Root Mean Square noise amplitude on a waveform at the mid reference level.

SIGMA1 (histogram measurement) measures the percentage of points in the histogram that are within one standard deviation of the histogram mean.

SIGMA2 (histogram measurement) measures the percentage of points in the histogram that are within two standard deviations of the histogram mean.

SIGMA3 (histogram measurement) measures the percentage of points in the histogram that are within three standard deviations of the histogram mean.

SIXSigmajit (histogram measurement) is six × RMSJitter.

SNRatio measures the signal-to-noise ratio. The signal-to-noise ratio is the amplitude of a noise rejection band centered on the mid level.

STDdev measures the standard deviation (Root Mean Square (RMS) deviation) of all acquired points within or on the histogram box.

UNDEFINED is the default measurement type, which indicates that no measurement type is specified. Once a measurement type is chosen, it can be cleared using this argument.

WAVEFORMS (waveform count) measures the number of waveforms used to calculate the histogram.

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)

This query-only command returns the units of the immediate measurement.

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)

This query-only command 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:SOUrce<x> command. Immediate measurements and annotations are not displayed on the screen.

NOTE. *A change to CH<x>:SCALe will not necessarily have taken affect if followed by this command.*

Group

Measurement

Syntax

MEASUrement:IMMed:VALue?

Related Commands

MEASUrement:IMMed:TYPe, MEASUrement:IMMed:SOUrce<x>, *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:MEAS<x>? (Query Only)

This query-only command returns all measurement parameters for the displayed measurement specified by x, which can range from 1 through 8. This query command is equivalent to selecting Measurement Setup from the Measure menu and viewing the Measurements table; then choosing the Time tab, clicking the Delay button and viewing the Delay Edge and Search Direction settings.

Group Measurement

Syntax `MEASUrement:MEAS<x>?`

Examples `MEASUREMENT:MEAS1?` might return `:MEASUREMENT:MEAS1:STATE 0;TYPE UNDEFINED;UNITS "V";SOURCE1 CH1;SOURCE2 CH1;SOURCE1:SIGTYPE PULSE;:MEASUREMENT:MEAS1:SOURCE2:SIGTYPE PULSE;:MEASUREMENT:MEAS1:DELAY:EDGE1 RISE;EDGE2 RISE; DIRECTION FORWARDS;:MEASUREMENT:MEAS1:REFLEVEL:METHOD PERCENT;ABSOLUTE:HIGH 0.0000;LOW 0.0000;MID1 0.0000;MID2 0.0000;:MEASUREMENT:MEAS1:REFLEVEL:PERCENT:HIGH 90.0000;LOW 10.0000;MID1 50.0000;MID2 50.0000;:MEASUREMENT:MEAS1:METHOD HISTOGRAM;NOISE HIGH.`

MEASUrement:MEAS<x>:COUNt? (Query Only)

This query-only command returns the number of values accumulated for this measurement since the last statistical reset. Some values might have been ignored because they generated an error. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax `MEASUrement:MEAS<x>:COUNt?`

Examples `MEASUREMENT:MEAS3:COUNT?` might return `:MEASUREMENT:MEAS3:COUNT 3.247000000E+03.`

MEASUrement:MEAS<x>:DELay? (Query Only)

This query-only command returns the delay measurement parameters for the measurement specified by <x>, which ranges from 1 through 8. Measurement parameters are presented in the following order: Edge1, Edge2, and Direction. This query command is equivalent to selecting Time from the Measure menu, choosing Delay and viewing the Delay Edge and Search Direction settings.

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:DELay?
```

Examples `MEASUREMENT:MEAS3:DELAY?` might return `:MEASUREMENT:MEAS3:DELAY:EDGE1 RISE;EDGE2 RISE;DIRECTION FORWARDS`.

MEASUrement:MEAS<x>:DELay:DIREction

This command sets or queries the starting point and direction that determines the delay "to" edge when taking a delay measurement. Use the MEASUrement:MEAS<x>:SOUrce<x> command to specify the waveform. This command is equivalent to selecting Time from the Measure menu, choosing Delay from the drop-down list and then clicking the desired Search Direction setting. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:DELay:DIREction {BACKWards|FORWards}
MEASUrement:MEAS<x>:DELay:DIREction?
```

Related Commands MEASUrement:MEAS<x>:SOUrce<x>

Arguments

`BACKWards` means that 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 that 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:MEAS1: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 backward.

MEASUrement:MEAS<x>:DELay:EDGE<x>

This command sets or queries the slope of the edge that is used for the delay "from" waveform (edge1) and the delay "to" waveform (edge2) when taking a delay measurement. Use the `MEASUrement:MEAS<x>:SOURCE2` command to specify the waveform. This command is equivalent to selecting Time from the Measure menu, choosing Delay from the drop-down list and then clicking the desired Delay Edge setting. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:DELay:EDGE<x> {FALL|RISe}
MEASUrement:MEAS<x>:DELay:EDGE<x>?
```

Related Commands MEASUrement:MEAS<x>:SOUrce<x>

Arguments

`FALL` specifies the falling edge of the waveform.

`RISE` specifies the rising edge of the waveform.

Examples

`MEASUREMENT:MEAS3:DELAY:EDGE2 RISE` specifies that the rising edge of the waveform be used for measurement 3.

`MEASUREMENT:MEAS1:DELAY:EDGE2?` might return `:MEASUREMENT:MEAS1:DELAY:EDGE2 FALL`, indicating that the falling edge of the waveform is being used for measurement 1.

MEASUrement:MEAS<x>:MAXimum? (Query Only)

This query-only command returns the maximum value found for this measurement since the last statistical reset. Measurements are specified by x, which ranges from 1 through 8.

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)

This query-only command returns the mean value accumulated for this measurement since the last statistical reset. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax `MEASUrement:MEAS<x>:MEAN?`

Examples `MEASUREMENT:MEAS1:MEAN?` might return `:MEASUREMENT:MEAS1:MEAN 514.71E-09.`

MEASUrement:MEAS<x>:METHod

This command specifies or queries the method used to calculate the 0% and 100% reference level.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:METHod {HIStogram|MINMax|MEAN}
MEASUrement:MEAS<x>:METHod?
```

Arguments `HIStogram` sets the high and low waveform levels statistically using a histogram algorithm.

`MINMax` sets the high and low waveform levels to MAX and MIN, respectively.

`MEAN` sets the high and low waveform levels to their mean.

Examples `MEASUREMENT:MEAS1:METHOD HISTOGRAM` specifies that the high and low reference levels are set statistically.

`MEASUREMENT:MEAS1:METHOD?` might return `:MEASUREMENT:MEAS1:METHOD MINMAX`, indicating that the reference levels are set to MIN and MAX.

MEASUrement:MEAS<x>:MINImum? (Query Only)

This query-only command returns the minimum value found for this measurement since the last statistical reset. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:MINImum?
```

Examples `MEASUREMENT:MEAS1:MINIMUM?` might return `:MEASUREMENT:MEAS1:MINIMUM 1.75E-09`.

MEASUrement:MEAS<x>:NOISe

This command sets or queries whether the noise measurement is made on the high or low level of the waveform. Sending this command is equivalent to selecting Ref Levs > Eye > Top Level or Base Level in the Comm tab of the Measurement Setup dialog box. The Eye section is displayed only if you have an eye-pattern or optical measurement defined. Measurements are specified by x, which ranges from 1 through 8.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:NOISe {HIGH|LOW}
MEASUrement:MEAS<x>:NOISe?
```

Arguments `HIGH` argument causes the measurement for noise to be taken at the high level of the waveform.

`LOW` argument causes the measurement for noise to be taken at the low level of the waveform.

Examples `MEASUREMENT:MEAS1:NOISE HIGH` causes the noise measurement to be made at the high level of the waveform for Measurement 1.

`MEASUREMENT:MEAS2:NOISE?` might return `:MEASUREMENT:MEAS2:NOISE LOW`, indicating that Measurement 2 will measure noise at the low level of the waveform.

MEASUrement:MEAS<x>:REFLevel? (Query Only)

This query-only command returns the current reference level parameters. It returns them in the following order: ABSOLUTE and then PERCENT for individual user measurements. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax `MEASUrement:MEAS<x>:REFLevel?`

Examples `MEASUREMENT:MEAS1:REFLEVEL?` might return `:MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:HIGH 1.7100E+00`, indicating that the reference level is set to 1.71 V.

MEASUrement:MEAS<x>:REFLevel:ABSolute:HIGH

This command sets or queries the high reference level, and is the upper reference level when MEASUrement:MEAS<x>:REFLevel:METHod is set to Absolute. Measurements are specified by x, which ranges from 1 through 8.

NOTE. *This command affects the results of rise and fall measurements*

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:REFLevel:ABSolute:HIGH <NR3>
MEASUrement:MEAS<x>:REFLevel:ABSolute:HIGH?
```

Related Commands MEASUrement:MEAS<x>:REFLevel:METHod

Arguments `<NR3>` is the high reference level in volts. The default is 0.0 V.

Examples `MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:HIGH 1.71` sets the high reference level to 1.71 V.

`MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:HIGH?` might return `:MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:HIGH 1.7100E+00`, indicating that the absolute high reference level is set to 1.71 V.

MEASUrement:MEAS<x>:REFLevel:ABSolute:LOW

This command sets or queries the low reference level, and is the lower reference level when MEASUrement:MEAS<x>:REFLevel:METHod is set to Absolute. Measurements are specified by x, which ranges from 1 through 8.

NOTE. *This command affects the results of rise and fall measurements.*

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:REFLevel:ABSolute:LOW <NR3>
MEASUrement:MEAS<x>:REFLevel:ABSolute:LOW?
```

Related Commands MEASUrement:MEAS<x>:REFLevel:METHod

Arguments <NR3> is the low reference level in volts. The default is 0.0 V.

Examples `MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:LOW 0.0 V` sets the low reference level to 0.0 V.

`MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:LOW?` might return `:MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:LOW 0.0000E+00`, indicating that the absolute low reference level is set to 0.0 V.

MEASUrement:MEAS<x>:REFLevel:ABSolute:MID<x>

This command sets or queries the mid reference level for the “to” waveform when taking a delay measurement, and is the 50% reference level when MEASUrement:MEAS<x>:REFLevel:METHod is set to Absolute. Mid1 sets the “from” waveform and Mid2 sets the “to” waveform when taking a delay measurement. Measurements are specified by x, which ranges from 1 through 8.

NOTE. *This command affects the results of period, frequency, delay, and all cyclic measurements.*

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:REFLevel:ABSolute:MID<x> <NR3>
MEASUrement:MEAS<x>:REFLevel:ABSolute:MID<x>?
```

Related Commands MEASUrement:MEAS<x>:REFLevel:METHod

Arguments <NR3> is the mid reference level in volts. The default is 0.0 V.

Examples MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:MID2 0.5 sets the mid reference level for the delay waveform to 0.5 V.

MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:MID2? might return :MEASUREMENT:MEAS1:REFLEVEL:ABSOLUTE:MID2 0.5000E+00, indicating that the absolute mid2 reference level is set to 0.5 V.

MEASUrement:MEAS<x>:REFLevel:METHod

This command specifies or queries the reference level units used for measurement calculations. This command is equivalent to selecting Reference Levels from the Measure menu and then choosing the desired reference level from the Units group box. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:REFLevel:METHod {ABSolute|PERCent}
MEASUrement:MEAS<x>:REFLevel:METHod?
```

Arguments ABSolute specifies that the reference levels are set explicitly using the MEASUrement:MEAS<x>: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:MEAS<x>:REFLevel:PERCent commands.

Examples MEASUREMENT:MEAS1:REFLEVEL:METHOD ABSOLUTE specifies that explicit user-defined values are used for the reference levels.

MEASUREMENT:MEAS1:REFLEVEL:METHOD? might return :MEASUREMENT:MEAS1:REFLEVEL:METHOD PERCENT, indicating that the reference level units used are calculated as a percent relative to HIGH and LOW.

MEASUrement:MEAS<x>:REFLevel:PERCent:HIGH

This command sets or queries the percentage (where 100% is equal to HIGH) used to calculate the high reference level when MEASUrement:MEAS<x>:REFLevel:METHod is set to Percent. Measurements are specified by x, which ranges from 1 through 8.

NOTE. *This command affects the results of rise and fall measurements.*

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:REFLevel:PERCent:HIGH <NR3>
MEASUrement:MEAS<x>:REFLevel:PERCent:HIGH?
```

Related Commands MEASUrement:MEAS<x>:REFLevel:METHod

Arguments <NR3> is the high reference level, ranging from 0 to 100%. The default high reference level is 90%.

Examples MEASUREMENT:MEAS1:REFLEVEL:PERCENT:HIGH 95 sets the high reference level to 95% of HIGH.

MEASUREMENT:MEAS1:REFLEVEL:PERCENT:HIGH? might return :MEASUREMENT:MEAS1:REFLEVEL:PERCENT:HIGH 90, indicating that the percentage high reference level is set to 90% of HIGH.

MEASUrement:MEAS<x>:REFLevel:PERCent:LOW

This command sets or queries the percentage (where 100% is equal to HIGH) used to calculate the low reference level when MEASUrement:MEAS<x>:REFLevel:METHod is set to Percent. This command is equivalent to selecting Reference Levels from the Measure menu, and then entering the Percentage Low Ref value. Measurements are specified by x, which ranges from 1 through 8.

NOTE. *This command affects the results of rise and fall measurements.*

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:REFLevel:PERCent:LOW <NR3>
MEASUrement:MEAS<x>:REFLevel:PERCent:LOW?
```

Related Commands MEASUrement:MEAS<x>:REFLevel:METHod

Arguments <NR3> is the low reference level, ranging from 0 to 100%. The default low reference level is 10%.

Examples MEASUREMENT:MEAS1:REFLEVEL:PERCENT:LOW 15 sets the high reference level to 15% of HIGH.

MEASUREMENT:MEAS1:REFLEVEL:PERCENT:LOW? might return :MEASUREMENT:MEAS1:REFLEVEL:PERCENT:LOW 10, indicating that the percentage high reference level is set to 10% of HIGH.

MEASUrement:MEAS<x>:REFLevel:PERCent:MID<x>

This command sets or queries the percentage (where 100% is equal to HIGH) used to calculate the mid reference level for the second waveform specified when MEASUrement:MEAS<x>:REFLevel:METHod is set to Percent. Mid1 specifies the "from" waveform and Mid2 specifies the "to" waveform for delay measurements. Measurements are specified by x, which ranges from 1 through 8.

NOTE. *This command affects the results of delay measurements.*

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:REFLevel:PERCent:MID<x> <NR3>
MEASUrement:MEAS<x>:REFLevel:PERCent:MID<x>?
```

Related Commands MEASUrement:MEAS<x>:REFLevel:METHod

Arguments <NR3> is the mid reference level, ranging from 0 to 100%. The default mid reference level is 50%.

Examples MEASUREMENT:MEAS1:REFLEVEL:PERCENT:MID2 40 sets the mid2 reference level to 40% of HIGH.

MEASUREMENT:MEAS1:REFLEVEL:PERCENT:MID2? might return :MEASUREMENT:MEAS1:REFLEVEL:PERCENT:MID2 45, indicating that the percentage mid2 reference level is set to 45% of HIGH.

MEASUrement:MEAS<x>:SOUrce<x>

This command sets or queries the source for all single channel measurements and specifies the reference source to measure "to" when taking a delay measurement or phase measurement. Measurements are specified by x, which ranges from 1 through 8. This command is equivalent to selecting Measurement Setup from the Measure menu, selecting a measurement type of either Phase or Delay, and then choosing the desired measurement source.

Tip: Source2 measurements apply only to phase and delay measurement types, which require both a target (Source1) and reference (Source2) source.

Group Measurement

Syntax

```
MEASUrement:MEAS<x>:SOUrce<x>
{CH<x>|MATH<y>|REF<x>|HIStogram}
MEASUrement:MEAS<x>:SOUrce<x>?
```

Related Commands MEASUrement:MEAS<x>:TYPe

Arguments

`CH<x>` is an input channel waveform. The x variable can be expressed as an integer ranging from 1 through 4.

`MATH<y>` is a math waveform. The y variable can be expressed as an integer ranging from 1 through 4.

`REF<x>` is a reference waveform. The x variable can be expressed as an integer ranging from 1 through 4.

`HIStogram` is a histogram. Histogram is valid only for source 1.

Examples

`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>:SIGType

This command sets or queries the type of input signal used for the specified measurement slot. MEAS<x> is 1 through 8 for the measurement slot using SOURCE<x>, 1 or 2.

To ensure accurate measurements, use this command to specify the input-signal type for the measurement source.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:SOUrce<x>:SIGType {PULse|EYE}
MEASUrement:MEAS<x>:SOUrce<x>:SIGType?
```

Arguments `PULSE` is for generic signals that are not associated with synchronous communications standards.

`EYE` is for synchronous-communication signals with NRZ-like characteristics (nonreturn to zero).

Examples `MEASUREMENT:MEAS5:SOURCE1:SIGTYPE PULSE` selects Pulse as the signal type associated with source 1 for measurement slot 5.

`MEASUREMENT:MEAS2:SOURCE1:SIGTYPE?` might return `:MEASUREMENT:MEAS2:SOURCE1:SIGTYPE EYE`, indicating that the signal type associated with source 1 of measurement slot 2 is EYE.

MEASUrement:MEAS<x>:STATE

This command sets or queries whether the specified measurement slot is computed and displayed. The measurement slot is specified by x, which ranges from 1 through 8. This command is equivalent to selecting Measurement Setup from the Measure menu and then clicking the Display button.

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<x> command. You define the measurement type using the MEASUrement:MEAS<x>:TYPe command.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:STATE {<NR1>|OFF|ON}
MEASUrement:MEAS<x>:STATE?
```

Related Commands MEASUrement:MEAS<x>:SOUrce<x>, MEASUrement:MEAS<x>:TYPe

Arguments `<NR1>` = 0 disables calculation and display of the specified measurement slot; any other value enables calculation and display of the specified measurement slot.

OFF disables calculation and display of the specified measurement slot.

ON enables calculation and display of the specified measurement slot.

Examples MEASUREMENT:MEAS2:STATE ON computes and displays the measurement defined as measurement 2.

MEASUREMENT:MEAS1:STATE? might return :MEASUREMENT:MEAS1:STATE 0, indicating that measurement defined for measurement slot 1 is disabled.

MEASUrement:MEAS<x>:STDdev? (Query Only)

This query-only command returns the standard deviation of values accumulated for this measurement since the last statistical reset. Measurements are specified by x, which ranges from 1 through 8.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:STDdev?
```

Examples MEASUREMENT:MEAS1:STDDEV? might return :MEASUREMENT:MEAS1:STDDEV 21.0E-12.

MEASUrement:MEAS<x>:TYPe

This command sets or queries the measurement type defined for the specified measurement slot. The measurement slot is specified by x, which ranges from 1 through 8. This command is equivalent to selecting Measurement Setup from the Measure menu and then choosing the desired measurement type.

Group Measurement

Syntax
```
MEASUrement:MEAS<x>:TYPe {AMPlitude|AREa|
BURst|CAREa|CMEan|CRMs|DELay|DISTDUty|
EXTINCTDB|EXTINCTPCT|EXTINCTRATIO|EYEHeight|
EYEWIdth|FALL|FREQuency|HIGH|HITs|LOW|
MAXimum|MEAN|MEDian|MINImum|NCROss|NDUty|
NOVershoot|NWIdth|PBASe|PCROss|PCTCROss|PDUty|
PEAKHits|PERIod|PHAse|PK2Pk|PKPKJitter|
PKPKNoise|POVershoot|PTOP|PWIdth|QFACtor|
RISe|RMS|RMSJitter|RMSNoise|SIGMA1|SIGMA2|
SIGMA3|SIXSigmajit|SNRatio|STDdev|UNDEFINED| WAVEFORMS}
MEASUrement:MEAS<x>:TYPe?
```

Arguments

AMPlitude 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.

Amplitude = High - 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.

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.

CMEan (cycle mean) measures the arithmetic mean over the first cycle in the waveform or the first cycle in the gated region.

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.

DELay measures the time between the middle reference (default = 50%) amplitude point of the source waveform and the destination waveform.

DISTDUty (duty cycle distortion) measures the time between the falling edge and the rising edge of the eye pattern at the mid reference level. It is the peak-to-peak time variation of the first eye crossing measured at the mid-reference as a percent of the eye period.

EXTINCTDB measures the extinction ratio of an optical waveform (eye diagram). Extinction Ratio (dB) measures the ratio of the average power levels for the logic High to the logic Low of an optical waveform and expresses the result in dB. This measurement only works for fast acquisition signals or a reference waveform saved in fast acquisition mode.

Extinction dB = 10 × (log 10 (High / Low)

EXTINCTPCT measures the extinction ratio of the selected optical waveform. Extinction Ratio (%) measures the ratio of the average power levels for the logic Low (off) to the logic (High) (on) of an optical waveform and expresses the result in percent. This measurement only works for fast acquisition signals or a reference waveform saved in fast acquisition mode.

Extinction % = 100.0 × (Low / High)

EXTINCTRATIO measures the extinction ratio of the selected optical waveform. Extinction Ratio measures the ratio of the average power levels for the logic High to the logic Low of an optical waveform and expresses the result without units. This measurement only works for fast acquisition signals or a reference waveform

saved in fast acquisition mode. Extinction ratios greater than 100 or less than 1 generate errors; low must be greater than or equal to 1 µW.

Extinction Ratio = (High / Low)

`EYEHeight` measures the vertical opening of an eye diagram in volts.

`EYEWidth` measures the width of an eye diagram in seconds.

`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%).

`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.

You can also limit the High measurement (normally taken over the entire waveform record) to a gated region on the waveform.

`HITS` (histogram hits) measures the number of points in or on the histogram box.

`LOW` measures the Low reference (0% level, sometimes called Baseline) of a waveform.

`MAXimum` finds the maximum amplitude. This value is the most positive peak voltage found. It is measured over the entire waveform or gated region. When histogram is selected with the MEASUrement:METHod command, the maximum measurement measures the voltage of the highest nonzero bin in vertical histograms or the time of the right-most bin in horizontal histograms.

`MEAN` amplitude measurement finds the arithmetic mean over the entire waveform or gated region. When histogram is selected with the MEASUrement:METHod command, the mean measurement measures the average of all acquired points within or on the histogram.

`MEDian` (histogram measurement) measures the middle point of the histogram box. Half of all acquired points within or on the histogram box are less than this value and half are greater than this value.

`MINImum` finds the minimum amplitude. This value is typically the most negative peak voltage. It is measured over the entire waveform or gated region. When histogram is selected with the MEASUrement:METHod command, the minimum measurement measures the lowest nonzero bin in vertical histograms or the time of the left-most nonzero bin in the horizontal histograms.

`NCROss` (timing measurement) measures the time from the trigger point to the first falling edge of the waveform or gated region. The distance (time) is measured at the middle reference amplitude point of the signal.

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.

Negative Duty Cycle = (Negative Width) / Period × 100%

NOVershoot (negative overshoot) finds the negative overshoot value over the entire waveform or gated region.

Negative Overshoot = (Low - Minimum) / Amplitude × 100%)

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.

PBASe measures the base value used in extinction ratio measurements.

PCROss (timing measurement) measures the time from the trigger point to the first positive edge of the waveform or gated region. The distance (time) is measured at the middle reference amplitude point of the signal.

PCTCROss measures the location of the eye crossing point expressed as a percentage of EYEHeight.

Crossing percent = 100 ×[(eye-crossing-point - PBASe)/(PTOP - PBASe)]

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.

Positive Duty Cycle = (Positive Width)/Period × 100%

PEAKHits measures the number of points in the largest bin of the histogram.

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. When histogram is selected with the MEASUrement:METHod command, the PK2Pk measurement measures the histogram peak to peak difference.

PKPKJitter measures the variance (minimum and maximum values) in the time locations of the cross point.

PKPKNoise measures the peak-to-peak noise on a waveform at the mid reference level.

POVershoot

The positive overshoot amplitude measurement finds the positive overshoot value over the entire waveform or gated region.

Positive Overshoot = (Maximum - High) / Amplitude ×100%

PTOT measures the top value used in extinction ratio measurements.

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.

QFACtor measures the quality factor. The Q factor is a figure of merit for an eye diagram, which indicates the vertical eye opening relative to the noise at the low and high logic levels. It is the ratio of the eye size to noise.

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%).

RMS amplitude measurement finds the true Root Mean Square voltage in the entire waveform or gated region.

RMSJitter measures the variance in the time locations of the cross point. The RMS jitter is defined as one standard deviation at the cross point.

RMSNoise measures the Root Mean Square noise amplitude on a waveform at the mid reference level.

SIGMA1 (histogram measurement) measures the percentage of points in the histogram that are within one standard deviation of the histogram mean.

SIGMA2 (histogram measurement) measures the percentage of points in the histogram that are within two standard deviations of the histogram mean.

SIGMA3 (histogram measurement) measures the percentage of points in the histogram that are within three standard deviations of the histogram mean.

SIXSigmajit (histogram measurement) is six × RMSJitter.

SNRatio measures the signal-to-noise ratio. The signal-to-noise ratio is the amplitude of a noise rejection band centered on the mid level.

STDdev measures the standard deviation (Root Mean Square (RMS) deviation) of all acquired points within or on the histogram box.

UNDEFINED is the default measurement type, which indicates that no measurement type is specified. Once a measurement type is chosen, it can be cleared using this argument.

WAVEFORMS (waveform count) measures the number of waveforms used to calculate the histogram.

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)

This query-only command returns the units associated with the specified measurement. Measurements are specified by x, which ranges from 1 through 8.

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)

This query-only command returns the value that is calculated for the measurement specified by <x>, which ranges from 1 through 8. This command is equivalent to selecting Display Statistics from the Measure menu and then choosing Value from the drop-down list to display *all* measurement values on-screen.

NOTE. *This is the displayed value in the on-screen display. If measurement statistics are enabled, a new value is calculated with every waveform. In addition, this value is updated about every 1/3 second. If you are acquiring a long acquisition record, the instrument can 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

This command sets or queries the method used to calculate the 0% and 100% reference level. This command is equivalent to selecting Reference Levels from the Measure menu and then choosing the desired Determine Base, Top From setting.

Group Measurement

Syntax

```
MEASUrement:METHod {HIStogram|MEAN|MINMax}
MEASUrement:METHod?
```

Related Commands MEASUrement:REFLevel:PERCent:HIGH, MEASUrement:REFLevel:PERCent:LOW, MEASUrement:REFLevel:PERCent:MID<x>,

Arguments

`HIStogram` sets the high and low reference levels to the most common values either above or below the mid point, depending on whether the high reference point or the low reference point is being defined. Because the statistical approach ignores short-term aberrations, such as overshoot or ringing, the histogram method is the best setting for examining pulses.

`MEAN` sets the high and low reference levels to the mean values using all values either above or below the midpoint, depending on whether it is defining the high or low reference level. The selection is best used for examining eye patterns.

`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 HISTOGRAM` specifies that the high and low reference levels are set statistically.

`MEASUREMENT:METHOD?` might return `:MEASUREMENT:METHOD MINMAX`, indicating that the reference levels are set to MIN and MAX.

MEASUrement:NOISe

This command sets or queries whether the noise measurement is made on the high or low level of the waveform. Sending this command is equivalent to selecting Ref Levs > Eye > Top Level or Base Level in the Comm tab of the Measurement

Setup dialog box. The Eye section is displayed only if you have an eye-pattern or optical measurement defined.

Conditions This command is only valid if the instrument has Option MTM is installed

Group Measurement

Syntax
```
MEASUrement:NOISe {HIGH|LOW}
MEASUrement:NOISe?
```

Arguments `HIGH` argument causes the measurement for noise to be taken at the high level of the waveform.

`LOW` argument causes the measurement for noise to be taken at the low level of the waveform.

Examples `MEASUREMENT:NOISEHIGH` causes the noise measurement to be made at the high level of the waveform.

`MEASUREMENT:NOISE?` might return `:MEASUREMENT:NOISE LOW`, indicating that the noise measurement will be made at the low level of the waveform.

MEASUrement:REFLevel:ABSolute:HIGH

This command sets or queries 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 and is equivalent to selecting Reference Levels from the Measure menu, and then entering the Absolute High Ref value.

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: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

This command sets or queries the low reference level, and is the lower reference level when MEASUrement:REFLevel:METHod is set to Absolute. This command affects the results of rise and fall measurements and is equivalent to selecting Reference Levels from the Measure menu, and then entering the Absolute Low Ref value.

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:ABSolute:LOW <NR3>
MEASUrement:REFLevel:ABSolute:LOW?
```

Related Commands MEASUrement:REFLevel:METHod, MEASUrement:IMMed:TYPe, MEASUrement:MEAS<x>:TYPe

Arguments <NR3> is the low reference level in volts. The default is 0.0 V.

Examples MEASUREMENT:REFLEVEL:ABSOLUTE:LOW 0.0 sets the low reference level to 0.0 V.

MEASUREMENT:REFLEVEL:ABSOLUTE:LOW? might return :MEASUREMENT:REFLEVEL:ABSOLUTE:LOW 0.0000E+00, indicating that the absolute low reference level is set to 0.0 V.

MEASUrement:REFLevel:ABSolute:MID<x>

This command sets or queries the mid reference level for the "from" waveform (mid1) and the "to" waveform (mid2) 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 period, frequency, delay, and all cyclic measurements and is equivalent to selecting Reference Levels from the Measure menu, and then entering the Absolute Mid2 Ref value.

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:ABSolute:MID<x> <NR3>
MEASUrement:REFLevel:ABSolute:MID<x>?
```

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:METHod

This command specifies or queries the reference level units used for measurement calculations. This command is equivalent to selecting Reference Levels from the Measure menu, and then choosing the desired reference level from the Units group box.

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

This command sets or queries the percent (where 100% is equal to HIGH) that is 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 and is equivalent to selecting Reference Levels from the Measure menu, and then entering the Percentage High Ref value.

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:PERCent:HIGH <NR3>
MEASUrement:REFLevel: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

This command sets or queries the percent (where 100% is equal to HIGH) that is 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 and is equivalent to selecting Reference Levels from the Measure menu, and then entering the Percentage Low Ref value.

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:PERCent:LOW <NR3>
MEASUrement:REFLevel:PERCent:LOW?
```

Related Commands MEASUrement:REFLevel:METHod, MEASUrement:IMMed:TYPe, MEASUrement:MEAS<x>:TYPe

Arguments <NR3> is the low reference level, ranging from 0 to 100%. The default low reference level is 10%.

Examples MEASUREMENT:REFLEVEL:PERCENT:LOW 15 sets the high reference level to 15% of HIGH.

MEASUREMENT:REFLEVEL:PERCENT:LOW? might return :MEASUREMENT:REFLEVEL:PERCENT:LOW 10, indicating that the percentage high reference level is set to 10% of HIGH.

MEASUrement:REFLevel:PERCent:MID<x>

This command sets or queries 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. Mid1 specifies the percent for the first waveform and Mid2 specifies the percent for the second waveform specified for delay measurements. This command affects the results of period, frequency, delay, and all cyclic measurements and is equivalent to selecting Reference Levels from the Measure menu, and then entering the Percentage Mid Ref value.

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:PERCent:MID<x> <NR3>
MEASUrement:REFLevel:PERCent:MID<x>?
```

Related Commands MEASUrement:REFLevel:METHod

Arguments <NR3> is the mid reference level, ranging from 0 to 100%. The default mid reference level is 50%.

Examples MEASUREMENT:REFLEVEL:PERCENT:MID2 40 sets the mid2 reference level to 40% of HIGH.

MEASUREMENT:REFLEVEL:PERCENT:MID2? might return :MEASUREMENT:REFLEVEL:PERCENT:MID2 45, indicating that the percentage mid2 reference level is set to 45% of HIGH.

MEASUrement:SOUrce<x>:SIGType

This command sets or queries the type of input signal used for measurement SOURCE<x>, 1 or 2.

To ensure accurate measurements, use this command to specify the input-signal type for the measurement source.

Conditions This command is valid only if the instrument has Option MTM is installed

Group Measurement

Syntax
```
MEASUrement:SOUrce<x>:SIGType {PULse|EYE}
MEASUrement:SOUrce<x>:SIGType?
```

Arguments `PULSE` is for generic signals that are not associated with synchronous communications standards.

`EYE` is for synchronous-communication signals with NRZ-like characteristics (nonreturn to zero).

Examples `MEASUREMENT:SOURCE1:SIGTYPE PULSE` selects Pulse as the signal type for source 1.

`MEASUREMENT:SOURCE1:SIGTYPE?` might return `:MEASUREMENT:SOURCE1:SIGTYPE EYE`, indicating that the signal type for source 1 is EYE.

MEASUrement:STATIstics:COUNt (No Query Form)

This command (no query form) clears existing measurement statistics from memory. This command is equivalent to selecting Measurement Setup from the Measure menu, selecting Statistics, and clicking the Reset button.

Group Measurement

Syntax
```
MEASUrement:STATIstics:COUNt {RESET}
```

Related Commands MEASUrement:STATIstics:MODe

Arguments `RESET` clears existing measurement statistics from memory.

Examples `MEASUREMENT:STATISTICS:COUNT RESET` resets the count of the statistics measurement to zero, clears the measurement statistics from memory, and restarts the calculation.

MEASUrement:STATIstics:MODe

This command controls the operation and display of measurement statistics. This command is equivalent to selecting Measurement Setup from the Measure menu, clicking the Statistics button and then choosing the desired Measurement Format.

Group Measurement

Syntax

```
MEASUrement:STATIstics:MODe
{OFF|ALL|VALUEMean|MINMax|MEANSTDdev}
MEASUrement:STATIstics:MODe?
```

Related Commands MEASUrement:STATIstics:WEIghting

Arguments

`OFF` turns off all measurements. This is the default value.

`ALL` turns on statistics and displays all statistics for each measurement.

`VALUEMean` turns on statistics and displays the value and the mean (μ) of each measurement.

`MINMax` turns on statistics and displays the min and max of each measurement.

`MEANSTDdev` turns on statistics and displays the mean and standard deviation of each measurement.

Examples

`MEASUREMENT:STATISTICS:MODE OFF` turns off statistic measurements.

`MEASUREMENT:STATISTICS:MODE?` might return `:MEASUREMENT:STATISTICS:MODE ALL`, indicating that measurement statistics are turned on and all statistics are being displayed for each measurement.

MEASUrement:STATIstics:WEIghting

This command sets or queries the time constant for mean and standard deviation statistical accumulations. This command is equivalent to selecting Measurement Setup from the Measure menu, clicking the Statistics button and entering the desired Weight n= value.

Group Measurement

Syntax

```
MEASUrement:STATIstics:WEIghting <NR1>
MEASUrement:STATIstics:WEIghting?
```

Related Commands MEASUrement:STATIstics:MODe

Arguments <NR1> is the time constant for the mean and standard deviation statistical accumulations.

Examples MEASUREMENT:STATISTICS:WEIGHTING 4 sets statistical weighting to 4.

MEASUREMENT:STATISTICS:WEIGHTING? might return :MEASUREMENT:STATISTICS:WEIGHTING 4, indicating that measurement statistics weighting is currently set to 4.

MULTiscope:CONFig

This command sets or queries the current MultiScope operational configuration. This command is for the Master instrument in the multi-instrument configuration.

NOTE. *Switching configurations results in a restart of the TekScope application, and may take several minutes to complete.*

Group Miscellaneous

Syntax MULTiscope:CONFig {ATI | TEKCONnect | TIMESYnc}

Arguments ATI sets the MultiScope system to use the ATI channels.

TEKCONNECT sets the MultiScope system to use the TekConnect channels.

TIMESYNC sets the MultiScope system to use the TekConnect channels in TimeSync mode. TimeSync mode is only available on some instruments. The Master instrument controls the instrument hardware such as the horizontal and trigger settings. Desired results are only achieved in Single Sequence acquisition mode. Data from all channels can be acquired using programmable interface commands.

Examples MULTISCOPE:CONFIG ATI sets your MultiScope system to use the ATI channels.

MULTiscope:CONFig? might return :MULTiscope:CONFig TEKCONNECT indicating that your MultiScope system is set to use the TekConnect channels.

MULTiscope:EXIT (No Query Form)

This command instructs the Configuration Managers (and active scope applications) across the entire MultiScope configuration to shutdown.

Group Miscellaneous

Syntax `MULTiscope:EXIT`

Examples `MULTISCOPE:EXIT` causes the Configuration Managers (and active scope applications) across the entire MultiScope configuration to shutdown.

MULTiscope:RESTART (No Query Form)

This command instructs the Configuration Managers (and active scope applications) across the entire MultiScope configuration to exit and restart themselves.

Group Miscellaneous

Syntax `MULTiscope:RESTART`

Examples `MULTISCOPE:RESTART` causes the Configuration Manager and active scope applications across the entire MultiScope configuration to exit and restart themselves.

MULTiscope:STATUS? (Query Only)

This query returns the current state of oscilloscopes in the MultiScope system.

Group Miscellaneous

Syntax `MULTiscope:STATUS?`

Returns If all of the oscilloscopes are running correctly then “All Extensions are Running” is returned. If one or more extensions have failed, “Failed Extensions are <extension letters>” is returned with the failing extensions noted.

Examples `MULTISCOPE:STATUS?` might return `:MULTISCOPE:STATUS "Failed Extensions are B"`, indicating that extension B failed.

NEWpass (No Query Form)

This command (no query form) 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>`

Related Commands PASSWord, *PUD

Arguments `<QString>` is the new password, which can contain up to 10 characters.

Examples `NEWPASS"mypassword"` creates a new password (mypassword) for accessing your protected data.

*OPC

This command 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. For a complete discussion of the use of these registers and the output queue, see Registers and Queues.

The *OPC command allows you to synchronize the operation of the instrument with your application program. For more information, see Synchronization Methods.

Table 2-47: Commands that Generate an OPC Message

Operation	Command
Single sequence acquisition	ACQuire:STATE ON or ACQuire:STATE RUN or ACQuire:STATE 1 (when ACQuire:STOPAfter is set to SEQuence)
Starts diagnostics	DIAg:EXECUTE
Calibration step	Refer to the Service Manual.

Table 2-47: Commands that Generate an OPC Message (cont.)

Operation	Command
Test execution	TEST
Recall setup	*RST, FACtory, TEKSecure
	*RCL, RECAll:SETUp

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.

OPCEXtended

This command sets or queries the behavior of OPC commands and queries. When enabled, operations referenced in the *OPC command description notify when their overlapped functionality has completed. When disabled, the operations notify as they have in the past (only once updated in the instrument state database).

Table 2-48: Command synchronization

Operation	PI sequence
Single sequence with ttOff	:ACQUIRE:STOPAFTER SEQUENCE :ACQUIRE:STATE 1;*OPC?;:WFMOUTPRE:XZERO?
Single sequence with Measurement Annotation	:ACQUIRE:STOPAFTER SEQUENCE;:MEASUREMENT:MEAS1:STATE 1;TYPE PK2PK :ACQUIRE:STATE 1;*OPC?;:MEASUREMENT:ANNOTATION:X1?
Single sequence with Cursors	:ACQUIRE:STOPAFTER SEQUENCE;:CURSOR:FUNCTION WAVEFORM;SOURCE CH1;STATE 1 :ACQUIRE:STATE 1;*OPC?
Single sequence with Math	:ACQUIRE:STOPAFTER SEQUENCE;:MATH1:DEFINE "Ch1*Ch2";:SELECT:MATH1 1 :ACQUIRE:STATE 1;*OPC?
Default setup followed by Save Waveform	*RST;*OPC? :SAVE:WAVEFORM CH1,REF1;*WAI :SELECT:REF1 1

Table 2-48: Command synchronization (cont.)

Operation	PI sequence
Math On during Acq Run mode	:HORIZONTAL:MODE MANUAL;RECORDLENGTH 2500000 :MATH1:DEFINE "CH1*CH1";:SELECT:MATH1 1 :DATA:ENCDG ASCII;SOURCE REF1;START 1;STOP 10 :SELECT:MATH1 0 {Wait a couple sec..longer in release mode?} :SELECT:MATH1 1;*WAI;:CURVE?
Save Math to Ref	:HORIZONTAL:MODE MANUAL;RECORDLENGTH 2500000 :MATH1:DEFINE "CH1*CH1";:SELECT:MATH1 1;*WAI; :SAVE:WAVEFORM MATH1,REF1;:SELECT:REF1 1 :DATA:ENCDG ASCII;SOURCE REF1;START 1;STOP 10 CURVE?
Trigger state	:ACQUIRE:STOPAFTER SEQUENCE :ACQUIRE:STATE 1;*OPC?;:TRIGGER:STATE?
Single sequence with Measurement	:ACQUIRE:STOPAFTER SEQUENCE;:MEASUREMENT:MEAS1:STATE 1;TYPE AMPLITUDE :ACQUIRE:STATE 1;*OPC?;:MEASUREMENT:MEAS1:VALUE?
Single sequence with Measurement on Math	:ACQUIRE:STOPAFTER SEQUENCE;:HORIZONTAL:MODE MANUAL;RECORDLENGTH 2500000 :MATH1:DEFINE "CH1*CH1";:SELECT:MATH1 1 :MEASUREMENT:MEAS1:STATE 1;TYPE AMPLITUDE;SOURCE MATH1 :ACQUIRE:STATE 1;*OPC?;:MEASUREMENT:MEAS1:VALUE?
Acq Count	*RST;*WAI;:ACQUIRE:NUMACQ?
Acq state after single sequence	:ACQUIRE:STOPAFTER SEQUENCE;STATE 1;*WAI;:ACQUIRE:STATE?

Group

Status and Error

Syntax

```
OPCEXtended {<NR1>|OFF|ON}
OPCEXtended?
```

Arguments

`ON` turns on extended OPC behavior.

`OFF` turns off extended OPC behavior.

`<NR1>` = 0 turns off extended OPC behavior; any other value turns on extended OPC behavior.

Examples

`OPCEXTENDED ON` turns on extended OPC behavior.

`OPCEXTENDED?` might return `:OPCEXTENDED 1` indicating that extended OPC behavior is turned on.

*OPT? (Query Only)

This query-only command returns a list of the options installed in your instrument.

Group Status and Error

Syntax `*OPT?`

Examples `*OPT?` might return `:3M:16 Meg Max 4M/CH`.

PASSWord (No Query Form)

This command (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 instrument is powered off, or until the FACtory command, the PASSWord command with no arguments, or the *RST command 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, which can contain 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

This command sets and queries 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 {<NR1>|OFF|ON}
*PSC?
```

Related Commands DESE, *ESE, FACtory, *RST, *SRE

Arguments <NR1> = 0 sets the power-on status clear flag to false, disables the power-on clear and allows the instrument 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 and prevents any SRQ assertion after power on.

OFF sets the power-on status clear flag to false, disables the power-on clear and allows the instrument to possibly assert SRQ after power-on.

ON sets the power-on status clear flag to true, enabling the power-on status clear and prevents any SRQ assertion after power on.

Examples *PSC0 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

This command sets or queries 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 100 characters.

<QString> is a string containing up to 100 characters.

Examples *PUD#229This instrument belongs to me stores the string “This instrument belongs to me” in the user protected data area.

*PUD? might return #221PROPERTY OF COMPANY X.

*RCL (No Query Form)

This command (no query form) restores the state of the instrument from a copy of the settings stored in memory (The settings are stored using the *SAV command). If ‘factory' is referenced (by specifying ‘0'), the factory default values will be restored.

This command is equivalent to RECAll:SETup and performs the same function as selecting Recall from the File menu, and then pressing the Setup button.

Group Save and Recall

Syntax *RCL <NR1>

Related Commands DELEte:SETUp, FACtory, *LRN?, RECAll:SETUp, *RST, *SAV, SAVe:SETUp

Arguments <NR1> specifies a setup storage location value ranging from 0 through 10. Using an out-of-range value causes an execution error.

Examples *RCL 3 restores the instrument from a copy of the settings stored in memory location 3.

RECAll:MASK (No Query Form)

This command (no query form) recalls a stored mask as the User mask.

Group Mask

Syntax RECAll:MASK <QString>

Related Commands SAVe:MASK

Arguments

<QString> is a quoted string that defines the file name and path from where the mask will be recalled. Input the file path using the form <drive>/<dir>/<filename>. <drive> and one or more <dir>s are optional. If you do not specify them, the instrument will read the mask from the default directory. <filename> stands for a file name of up to 128 characters (use of wildcard characters in filenames is not supported). File name extensions are not required but are highly recommended.

The file is stored in ASCII format and is user-editable.

Examples

RECALL:MASK "TEST" recalls the mask stored in the file named TEST as the User mask.

RECAll:SETUp (No Query Form)

This command (no query form) returns stored or factory settings to the instrument from a copy of the settings stored in memory. If factory is referenced (by specifying '0'), the factory default values will be restored. This command is equivalent to *RCL and performs the same function as selecting Recall from the File menu, and then choosing the Setup button.

Group

Save and Recall

Syntax

RECAll:SETUp {FACtory|<NR1>|<file path>}

Related Commands

FACtory, *RCL, *RST, *SAV, SAVe:SETUp

Arguments

FACtory restores the factory setup.

<NR1> specifies a setup storage location to restore. Setup storage location values range from 0 through 10; using an out of range value causes an error (222, "Data out of range").

<file path> specifies a location for an instrument 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>.<drive> and one or more <dir>s are optional. If you do not specify them, the instrument will read the file from the default directory (C:\Users\Public\Tektronix\TekScope\setups). <filename> stands for a file name of up to 128 characters (use of wildcard characters in filenames is not supported). File name extensions are not required, but highly recommended.

Examples

RECALL:SETUP FACTORY recalls (and makes current) the instrument setup to its factory defaults.

RECALL:SETUP 2 recalls the instrument setup from setup storage location 2.

RECALL:SETUP "TEK00000.SET" recalls the setup from the file TEK00000.SET in the default directory for setups (C:\Users\Public\Tektronix\TekScope\setups).

RECAll:SETUp:DESKew

This command sets or queries the deskew values that are affected by a default setup or a recalled setup.

NOTE. *This setting is changed only by TekSecure (which resets it to the default setting of UNLOCK) or through explicit user request (from the Recall Setup dialog box in the TekScope User Interface).*

Group Save and Recall

Syntax

```
RECAll:SETUp:DESKew {LOCK|UNLOCK}
RECAll:SETUp:DESKew?
```

Arguments

LOCK a default or a recall setup will not change the deskew settings when this option is selected.

UNLOCK a default setup will reset the deskew settings to factory values and a recall setup will apply the setup deskew values to the recalled oscilloscope state.

Examples

RECALL:SETUP:DESKEW LOCK, a default or recall setup will not change the deskew settings.

RECALL:SETUP:DESKEW? might return :RECALL:SETUP:DESKEW UNLOCK, indicating that a default setup will reset the deskew settings to factory values and a recall setup will apply the setup deskew values to the recalled oscilloscope state.

RECAll:WAVEform (No Query Form)

This command (no query form) recalls a stored waveform to a reference location. This command is equivalent to selecting Recall from the File menu, and then pressing the Waveform button.

Group Save and Recall

Syntax

```
RECAll:WAVEform <file path>,REF<x>
```

Related Commands DELEte:WAVEform, SAVe:WAVEform

Arguments <file path> specifies a location for a stored waveform file. <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>:<dir>/<filename>.<drive> and one or more <dir>s are optional. If you do not specify them, the instrument will read the waveform from the default directory (C:\Users\Public\Tektronix\TekScope|Waveforms). <filename> stands for a file name of up to 128 characters (use of wildcard characters in filenames is not supported). File name extensions are not required but are highly recommended.

REF<x> specifies a location in internal reference memory. Reference memory location values range from 1 through 4.

Examples RECALL:WAVEFORM "TEK00000.WFM",REF1 recalls the waveform stored in the file named TEK00000.WFM from the default directory for waveforms (C:\Users\Public\Tektronix\TekScope\Waveforms) to the reference location 1.

REF<x>:HORizontal:POSition

This command sets or queries the horizontal display position of the reference waveform. The reference waveform is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Reference Waveforms from the File menu, choosing Reference Setup¼ from the drop-down list, selecting a reference waveform, and then entering the horizontal position value using the multipurpose knob.

Group Vertical

Syntax
```
REF<x>:HORizontal:POSition <NR3>
REF<x>:HORizontal:POSition?
```

Arguments <NR3> argument specifies the horizontal position of the specified reference waveform in percentage of the waveform that is displayed to the right of the center vertical graticule. The range of this argument is from 0 through 100.

Examples REF2:HORIZONTAL:POSITION 0.0 positions the start of the Reference 2 waveform at the center graticule.

REF1:HORIZONTAL:POSITION? might return :REF1:HORIZONTAL:POSITION 50, indicating that the Reference 1 waveform is centered on the horizontal graticule.

REF<x>:LABel? (Query Only)

This query-only command returns a branch query containing the waveform label name and the coordinates at which the label (attached to the displayed waveform of the specified reference) is displayed. The reference waveform is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Reference Waveforms from the File menu and then choosing Label from the drop-down list.

Group Vertical

Syntax
```
REF<x>:LABel?
```

Related Commands REF<x>:LABel:NAMe, REF<x>:LABel:XPOS, REF<x>:LABel:YPOS

Examples `REF1:LABEL?` might return `:REF1:LABel:NAME "Myname";:XPOS-200;:YPOS 50`.

REF<x>:LABel:NAMe

This command sets or queries the label of the designated waveform. The reference waveform is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Reference Waveforms from the File menu, choosing Label from the drop-down list, selecting the tab associated with the reference for which you want to create a label, and entering a label in the Label box.

Group Vertical

Syntax
```
REF<x>:LABel:NAMe <QString>
REF<x>:LABel:NAMe?
```

Related Commands REF<x>:LABel?

Arguments `<QString>` is the character string that will be used for the reference waveform label name.

Examples `REF4:LABEL:NAME "My Reference"` sets the label name of Reference 4 waveform to “My Reference”.

REF3:LABEL:NAME? might return :REF3:LABEL:NAME "Signal2", indicating that the label name for Reference 3 waveform is currently set to “Signal2”.

REF<x>:LABel:XPOS

This command sets or queries the X screen offset at which the label (attached to the displayed waveform of the specified reference) is displayed, relative to the left edge of the screen The reference waveform is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Reference Waveforms from the File menu, choosing Label from the drop-down list, selecting the tab associated with the reference for which you want to position a label, and entering a value in the X Position box.

Group Vertical

Syntax

```
REF<x>:LABel:XPOS <NR1>
REF<x>:LABel:XPOS?
```

Related Commands REF<x>:LABel?

Arguments <NR1> is the location (control in divisions) where the waveform label for the selected reference is displayed, relative to the left edge of the screen. Arguments should be integers ranging from 0 through 10.

Examples REF4:LABEL:XPOS 10 moves the waveform label for the Reference 3 waveform, so that it begins 10 divisions to the right of the left edge of the screen.

REF2:LABEL:XPOS? might return :REF2:LABEL:XPOS 1.5, indicating that the x-axis for the Reference 2 waveform is currently 1.5 divisions to the right of the left edge of the screen.

REF<x>:LABel:YPOS

This command sets or queries the Y screen offset at which the label (attached to the displayed waveform of the specified reference) is displayed, relative to the center of the screen. The reference waveform is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Reference Waveforms from the File menu, choosing Label from the drop-down list, selecting the tab associated with the reference for which you want to position a label, and entering a value in the Y Position box.

Group Vertical

Syntax
```
REF<x>:LABel:YPOS <NR1>
REF<x>:LABel:YPOS?
```

Related Commands REF<x>:LABel?

Arguments <NR1> is the location (in divisions) where the waveform label for the selected reference is displayed, relative to the waveform handle. Arguments should be integers ranging from 10 to -10.

Examples REF3:LABEL:YPOS -10 moves the waveform label for the Reference 3 waveform 10 divisions below the waveform handle.

REF2:LABEL:YPOS? might return :REF2:LABEL:YPOS 0, indicating that the waveform label for the Reference 2 waveform is currently located at the waveform handle.

REF<x>:THRESHold

This command sets or queries the comparable threshold for converting the reference signal to digital form for the channel specified by x. The value of x can range from 1 through 4.

Group Digital

Syntax
```
REF<x>:THRESHold <NR3>
REF<x>:THRESHold?
```

Arguments <NR3> specifies the reference threshold in volts.

Examples REF1:THRESHOLD 1 sets the threshold value of the reference REF1 to 1 volt.

REF1:THRESHOLD? might return :REF1:THRESHOLD 500.0000E-3, indicating that the threshold value set for the reference REF1 is 0.5 volts.

REF<x>:VERTical:POSition

This command sets or queries the vertical position of a reference waveform The reference waveform is specified by x. The value of x can range from 1 through 4.

This command is equivalent to selecting Reference Waveforms from the File menu, choosing Reference Setup from the drop-down list, selecting a reference waveform, and then entering the Position value using the multipurpose knob.

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.

NOTE. *References are static. All position and scale changes are applied in a post-processing mode.*

Group Vertical

Syntax

```
REF<x>:VERTical:POSition <NR3>
REF<x>:VERTical:POSition?
```

Related Commands CH<x>:POSition, MATH<x>:VERTical:POSition, D<x>:POSition

Arguments <NR3> is the desired position, control in divisions from the center horizontal graticule. The range is from -5 to 5 divisions.

Examples `REF2:VERTICAL:POSITION 1.3E+00` positions the Reference 2 waveform 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 waveform is 1.3 divisions below the center horizontal graticule.

REF<x>:VERTical:SCAle

This command sets or queries the vertical scale of a reference waveform. The reference waveform is specified by x. The value of x can range from 1 through 4. This command is equivalent to selecting Reference Waveforms from the File menu, choosing Reference Setup from the drop-down list, selecting a reference waveform, and then entering the Scale value using the multipurpose knob.

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 and 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<x>:VERTical:SCAle

Arguments <NR3> is the gain in user units per division.

Examples REF4:VERTICAL:SCALE 100E-03 sets the Reference 4 waveform scale to 100 mV per division.

REF4:VERTICAL:SCALE? might return :REF2:VERTICAL:SCALE 1.0000e+00, indicating that the current vertical scale setting for Reference 2 waveform is 1 V per division.

REM (No Query Form)

This command (no query form) embeds a comment within GPIB programs as a means of internally documenting the programs. The instrument ignores these embedded comment lines.

Group Miscellaneous

Syntax
```
REM <QString>
```

Related Commands NEWpass, *PUD

Arguments <QString> is a string that can contain a maximum of 80 characters.

Examples `REM "This is a comment"` is a comment string that the instrument will ignore.

ROSc:OUT:FREQuency

NOTE. *Not available on some models.*

This command sets or returns the selected frequency for the timebase reference output signal.

Group Miscellaneous

Syntax
```
ROSc:OUT:FREQuency {MHZ10|MHZ100}
ROSc:OUT:FREQuency?
```

Arguments `MHZ10` outputs a 10 MHz reference signal to the REF OUT connector.

`MHZ100` outputs a 100 MHz reference signal to the REF OUT connector.

Examples `ROSC:OUT:FREQUENCYMHZ10` outputs a 10 MHz reference signal to the REF OUT connector.

`ROSC:OUT:FREQUENCY?` might return `ROSC:OUT:FREQUENCY MHZ100` indicating that the 100 MHz reference signal is output to the REF OUT connector.

ROSc:OUT:ULTRAsync

This command sets or queries the state of the UltraSync 12.5 GHz Clock Out. DPO70000SX Series only.

Group Miscellaneous

Syntax
```
ROSc:OUT:ULTRAsync {OFF|ON}
ROSc:OUT:ULTRAsync?
```

Arguments `OFF` disables the 12.5 GHz Clock Out.

`ON` enables the 12.5 GHz Clock Out.

Examples

ROSC:OUT:ULTRASYNC ON enables the UltraSync 12.5 GHz Clock Out.

ROSC:OUT:ULTRASYNC? might return ROSC:OUT:ULTRASYNC 0 indicating the UltraSync 12.5 GHz Clock Out is disabled.

ROSc:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the selected source for the time base reference oscillator. The reference oscillator locks to this source. Depending on the command argument that you specify, you can use an external reference or use the internal crystal oscillator as the time base reference.

This command is also useful for synchronizing multiple instruments.

Group

Miscellaneous

Syntax

```
ROSc:SOUrce {ULTRAsync|INTERnal|EXTernal}
ROSc:SOUrce {INTERnal|EXTernal|TEKLink}
ROSc:SOUrce?
```

Related Commands

ROSc:STATE?

Arguments

ULTRAsync specifies the 12.5 GHz UltraSync Clock as the time base reference.

INTERnal specifies the internal 10 MHz crystal oscillator as the time base reference.

EXTernal specifies the user-supplied external signal as the time base reference.

TEKLink specifies the TekLink signal as the time base reference.

Examples

ROSC:SOURCE INTERNAL specifies the internal 10 MHz crystal oscillator as the time base reference.

ROSC:SOURCE? might return :ROSC:SOURCE INTERNAL, indicating that the 10 MHz crystal oscillator is being used as the time base reference.

ROSc:STATE? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns whether the time base reference oscillator is locked. This command will return either LOCKED or UNLOCKED.

Group Miscellaneous

Syntax `ROSc:STATE?`

Related Commands ROSc:TRACking

Examples `ROSC:STATE?` might return `:ROSC:STATE LOCKED`, indicating that the time base reference is locked.

ROSc:TRACking

NOTE. *Not available on some models.*

This command sets or queries the tracking mode for the time base reference oscillator. The reference oscillator locks to the source. Depending on the command argument that you specify, you can use an external reference signal that is fed through or bypasses the phase-locked loop.

This command is also useful for synchronizing multiple instruments.

Group Miscellaneous

Syntax
```
ROSc:TRACking {STABle|FAST}
ROSc:TRACking?
```

Arguments `STABle` tracking mode specifies that the external reference signal is fed through a phase-locked loop that removes jitter from the external reference.

`FAST` tracking mode specifies that the external reference signal bypasses the phase-locked loop.

Examples `ROSC:TRACKING STABLE` feeds the external reference signal through the instrument phase-locked loop.

`ROSC:TRACKING?` might return `:ROSC:TRACKING FAST`, indicating that the external reference signal bypasses the phase-locked loop.

*RST (No Query Form)

This command (no query form) resets the instrument to the factory default settings. The *RST command does not change the following:

- The state of the IEEE Std 488.1-1987 interface
- The selected IEEE Std 488.1-1987 address of the instrument
- 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.

*RST only resets the programmable interface settings, it does not change the user interface settings.

Group Status and Error

Syntax `*RST`

Related Commands FACtory, RECAll:SETUp, SAVe:SETUp

Arguments None

Examples `*RST` resets the instrument settings to factory defaults.

*SAV (No Query Form)

This command (no query form) stores the state of the instrument to a specified memory location. You can later use the *RCL command to restore the instrument to this saved state.

This command is equivalent to selecting Save As from the File menu, and then pressing the Setup button.

Group Save and Recall

Syntax `*SAV <NR1>`

Related Commands *RCL, RECAll:SETUp, SAVe:SETUp

Arguments <NR1> specifies a location in which to save the state of the instrument. 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.

Examples `*SAV 2` saves the current instrument state in memory location 2.

SAVe:EVENTtable:BUS<x> (No Query Form)

Saves the event table contents of the specified bus in CSV format, to the file that you specify by <string>.

Group Save and Recall

Syntax `SAVe:EVENTtable:BUS<x> <string>`

Arguments <string> is the file name.

Examples `SAVE:EVENTTABLE:BUS2 "EVENTTABLE"` stores the event table for bus B2 in a file named "`EVENTTABLE`", to the default location `C:\users\username\Tektronix\TekScope`.

SAVe:MARKS (No Query Form)

This command saves the user search marks in CSV format, to the file that you specify by <string>.

Group Save and Recall

Syntax `SAVe:MARKS <string>`

Arguments <string> specifies the file that the marks are save to.

Examples `SAVE:MARKS “C:\users\username\Tektronix\TekScope\mymarks\mymarks.csv”` sets that path and file name as the place to store the search marks.

SAVe:MASK (No Query Form)

This command (no query form) saves the user mask to a mass storage file.

Group Mask

Syntax `SAVe:MASK <QString>`

Related Commands RECAll:MASK

Arguments `<QString>` is the location to where the mask is stored. <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>/<dir>/<filename>. <drive> and one or more <dir>s are optional.

The file is stored in ASCII format and is user-editable.

Examples `SAVE:MASK "TEST"` saves the user mask to the file TEST in the default directory and on the default drive.

SAVe:SETUp (No Query Form)

This command (no query form) stores the state of the instrument to a specified memory location. You can later use the *RCL command to restore the instrument to this saved state. This is equivalent to selecting Save As from the File menu, and then pressing the Setup button.

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>. <drive> and one or more <dir>s are optional. If you do not specify them, the instrument will store the file in the default directory (C:\Users\Public\Tektronix\TekScope\setups). <filename> stands for a file name of up to 128 characters. (Use of wildcard characters in filenames is not supported.) File name 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 instrument setup in memory location 5.

SAVE:SETUP"TEK00000.SET" saves the current instrument setup in the file TEK00000.SET in the default directory for setups (C:\Users\Public\Tektronix\TekScope\setups).

SAVe:WAVEform (No Query Form)

This command (no query form) saves a waveform to one of four reference memory locations or a file. This command is equivalent to selecting Reference Waveforms from the File menu and choosing Save Wfm from the drop-down list.

Group

Save and Recall

Syntax

```
SAVe:WAVEform [<wfm>,REF<x>]| [<wfm>,<QString>]
| [ALL,[QString]]| [DIGITALALL,<QString>]
```

Related Commands

DELEte:WAVEform, RECAll:WAVEform, SAVe:WAVEform:FILEFormat, SAVe:WAVEform:DATa:STARt, SAVe:WAVEform:DATa:STOP

Arguments

<wfm>,REF<x> saves the specified waveform to the specified reference memory location. <wfm> can be any of CH<x>, MATH<x>, and REF<x>.

<wfm>,<QString> saves the specified waveform to the file specified. The file is a quoted string that defines the file name and path. Input the file path using the form <drive>/<dir>/<filename>. <drive> and one or more <dir>s are optional. If you do not specify them, the instrument will write the waveform to the default directory. <filename> stands for a file name of up to 125 characters, followed by a period (".") and the three-character extension. Internal format waveforms use

the .wfm extension, spreadsheet format files use the .csv extension and MathCad format files use the .dat extension.

`ALL,[QString>]` saves all displayed waveforms, excluding digital waveforms, to individual files based on the `SAVE:WAVEFORM:FILEFORMAT` setting. If a file name argument is specified, it is used as a prefix for each file followed by the waveform ID (such as CH1, MATH1, REF1). If no file name is specified, the default prefix "TDSWfm_" is used, and the file is saved to the default directory.

`DIGITALALL,<QString>` saves all digital waveforms to a single file specified by the quoted string argument and based on the `SAVE:WAVEFORM:FILEFORMAT` setting. Possible output formats include `SPREADSHEET.CSV`. `SPREADSHEET.TXT`, and `INTERNAL`. Any other formats default to `SPREADSHEET.CSV` output.

Examples

`SAVE:WAVEFORM MATH2,REF2` saves the Math2 waveform in reference memory location2.

`SAVE:WAVEFORM MATH1,"TEK0000.WFM"` saves the Math1 waveform to the file TEK00000.WFM in the default directory and on the default drive.

SAVe:WAVEform:DATa:STARt

Sets or queries the save waveform starting waveform data sample.

NOTE. *Save:Waveform:FileFormat Auto always uses the entire waveform record and does not utilize SAVe:WAVEform:DATa:STARt or STOP or any other limitations on output data.*

Group

Save and Recall

Syntax

`SAVe:WAVEform:DATa:STARt {<NR1>}`

Related Commands

SAVe:WAVEform, SAVe:WAVEform:FILEFormat, SAVe:WAVEform:DATa:STOP

Arguments

`<NR1>` specifies the starting waveform sample used by the SAVe:WAVEform command. This works for Spreadsheet CSV, Spreadsheet TXT, Mathcad, and Matlab file formats.

Examples

`SAVE:WAVEFORM:DATA:START 150` specifies that the saved waveform data starts at waveform sample 150.

SAVE:WAVEFORM:DATA:START? might return SAVE:WAVEFORM:DATA:START 200, indicating that first data point of the saved waveform is waveform sample 200.

SAVe:WAVEform:DATa:STOP

Sets or queries the save waveform ending waveform data sample.

NOTE. *Save:Waveform:FileFormat Auto always uses the entire waveform record and does not utilize SAVe:WAVEform:DATa:STARt or STOP or any other limitations on output data.*

Group Save and Recall

Syntax SAVe:WAVEform:DATa:STOP {<NR1>}

Related Commands SAVe:WAVEform, SAVe:WAVEform:FILEFormat, SAVe:WAVEform:DATa:STARt

Arguments <NR1> specifies the ending waveform sample used by the SAVe:WAVEform command. This works for Spreadsheet CSV, Spreadsheet TXT, Mathcad, and Matlab file formats.

Examples SAVE:WAVEFORM:DATA:STOP 500 specifies that the saved waveform data stops at waveform sample 500.

SAVE:WAVEFORM:DATA:STOP? might return SAVE:WAVEFORM:DATA:STOP 1000, indicating that the last data point of the saved waveform is waveform sample 1000.

SAVe:WAVEform:FILEFormat

This command 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 instrument saves DPO waveforms as a 1000 x 502 matrix, with the first row corresponding to the most recently acquired data. The values specified by SAVe:WAVEform:DATa:STARt and SAVe:WAVEform:DATa:STOP determine the range of waveform data to output. In the event that SAVe:WAVEform:DATa:STOP value is greater than the current record length, the current record length determines the last output value.

This command is equivalent to selecting Save As from the File menu, clicking the Waveform button, and selecting the desired waveform file format.

Group Save and Recall

Syntax

```
SAVe:WAVEform:FILEFormat
{INTERNal|MATHCad|MATLab|SPREADSHEETCsv|
SPREADSHEETTxt|TIMEStamp|AUTO}
SAVe:WAVEform:FILEFormat?
```

Related Commands CURVe, DATa, DATa:STARt, DATa:STOP, SAVe:WAVEform, WFMInpre:NR_Pt, WFMOutpre:NR_Pt?, SAVe:WAVEform:DATa:STARt, SAVe:WAVEform:DATa:STOP

Arguments

`INTERNal` specifies that waveforms are saved in an internal format, using a .wfm file name extension. These files can be recalled as reference waveforms. When this argument is specified, the settings specified via the SAVe:WAVEform:DATa:STARt and SAVe:WAVEform:DATa:STOP commands have no meaning as the entire waveform is saved.

`MATHCad` specifies that waveforms are saved in MathCad format, using a .dat file name extension. When saving in this format, waveform values are delimited with new lines. MathCad format enables easy import of waveform data into MathCad or MATLAB. For FastAcq waveforms, data is imported as a matrix. For these formats, waveform header and timing information is saved in a separate header file. MathCad format header files use a _hdr.dat extension.

`MATLab` specifies that waveforms are saved in Matlab format, using a .dat file name extension. When saving in this format, waveform values are delimited with new lines. MATLAB format enables easy import of waveform data into MathCad or MATLAB. For FastAcq waveforms, data is imported as a matrix. For these formats, waveform header and timing information is saved in a separate header file. MATLAB format header files use a _hdr.dat extension.

`SPREADSHEETCsv` specifies that waveform data is saved in a format that contains comma delimited values. These waveform data files are named using the .csv file name extension. Saving waveforms in CSV format enables spreadsheet programs to import the data.

`SPREADSHEETTxt` specifies that waveform data is saved in a format that contains tab delimited values. These waveform data files are named using the .txt file name extension. Saving waveforms in this format enables spreadsheet programs to import the data.

`TIMEStamp` specifies that timestamp data is saved in a format that contains comma delimited values. These data files are named using the .txt file name extension. Saving waveforms in this format enables spreadsheet programs to import the data.

AUTO specifies that the file format should be taken from the file name extension. Supported extensions include *.wfm, *.csv, *.txt, *.h5). If an extension is read that is not supported, no file will be written out. If no extension is given in the filename, *.csv will be written out. Auto format always uses the entire waveform record and does not utilize SAVe:WAVEform:DATa:STARt or STOP or any other limitations on output 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:FORCESAMEFilesize

This command sets or queries the save waveform force same file size.

Group

Save and Recall

Syntax

```
SAVe:WAVEform:FORCESAMEFilesize {<NR1>|OFF|ON}
SAVe:WAVEform:FORCESAMEFilesize?
```

Arguments

<NR1> = 0 disables the function; any other value enables it.

OFF disables the function.

ON enables the function.

Examples

SAVe:WAVEform:FORCESAMEFilesize ON forces the same files size for waveform saves.

SAVe:WAVEform:FORCESAMEFilesize? might return :SAVE:WAVEFORM:FORCESAMEFILESIZE 0, indicating that force same file size is disabled.

SAVEON (No Query Form)

Sets the auto-increment file count to 0.

Once the number of saved files has reached the limit that you set (using the SAVEON:NUMevents command), no files will be saved until you reset the count.

Group

Save On

Syntax `SAVEON {RESET}`

Related Commands SAVEON:NUMEvents

Arguments `RESET` sets the file count to 0.

Examples `SAVEON RESET` sets the file count to 0.

SAVEON:COUNt? (Query Only)

Returns the number of events (files) that have been saved since the last reset.

Group Save On

Syntax `SAVEON:COUNt?`

Related Commands SAVEON, SAVEON:NUMEvents,

Examples `SAVEON:COUNT?` might return `:SAVEON:COUNT 15`, indicating that 15 events have been saved.

SAVEON:FILE:AUTOInc

Sets or queries the state of the auto increment file name feature (on or off). If AUTOInc is on, each time that an image, measurement, or waveform is saved to a file, the number that is appended to the file name will be augmented automatically, depending on the settings that you have made in the Related Commands, below. If AUTOInc is off, the previous file will be overwritten.

Group Save On

Syntax
```
SAVEON:FILE:AUTOInc {<NR1>|ON|OFF}
SAVEON:FILE:AUTOInc?
```

Related Commands SAVEON:COUNt?, SAVEON:FILE:COUNt, SAVEON:FILE:DEST, SAVEON:FILE:NAME, SAVEON:FILE:TYPE, SAVEON:NUMEvents

Arguments

<NR1> = 1 enables the Auto increment function; any other value disables the function.

ON indicates that each time that an image, measurement, or waveform is saved, the number that is appended to the file name will be augmented automatically.

OFF indicates that when an image, measurement, or waveform file is saved, the name will not be augmented and the previously saved file will be overwritten.

Examples

SAVEON:FILE:AUTOINC ON turns on the auto increment feature.

SAVEON:FILE:AUTOINC? might return :SAVEON:FILE:AUTOINC 1, indicating that the auto increment feature is enabled.

SAVEON:FILE:COUNt

Sets or queries the starting number that will be appended to the custom file name when SAVEON:FILE:AUTOInc is enabled and an image, measurement, or waveform is saved. The total of files saved cannot exceed 32767. If this file count is reached, files will not be saved until you change the file count to a lower number.

Group

Save On

Syntax

```
SAVEON:FILE:COUNt <NR3>
SAVEON:FILE:COUNt?
```

Related Commands

SAVEON:FILE:AUTOInc, SAVEON:FILE:TYPE, SAVEON:NUMEvents

Arguments

<NR3> specifies the starting number for automatically incrementing the file name.

Examples

SAVEON:FILE:COUNT250 specifies that the number to be appended to the file name will start with 250.

SAVEON:FILE:COUNT? might return :SAVEON:FILE:COUNT 500, indicating that when the oscilloscope starts saving files, it will start with 500 as the number that is appended to the end of the file name. It cannot go above 32767.

SAVEON:FILE:DEST

Sets or queries the location where files are saved. The default destination is C:\users\username\Tektronix\TekScope\SaveOnTrigger\. You can save the files to a local drive or network path by entering the desired location in <string>. You

can also select to save the files to a USB drive, using the browse button in the oscilloscope interface.

Group Save On

Syntax
```
SAVEON:FILE:DEST <string>
SAVEON:FILE:DEST?
```

Related Commands SAVEON:FILE:NAME

Arguments <string> specifies the location to store files.

Examples `SAVEON:FILE:DEST "C:\users\username\Tektronix\TekScope\SaveOnTrigger"` sets this as the location to save files (named by the `SAVEON:FILE:NAME` command), when there is a trigger.

`SAVEON:FILE:DEST?` might return `:SAVEON:FILE:DEST "C:\users\username\Tektronix\TekScope\SaveOnEvent"`, indicating the drive location where files will be saved when there is a trigger.

SAVEON:FILE:NAME

Sets or queries the file name to use when the file type is set to Custom.

Selecting a file name. You can select your own file name by entering the desired name in the <string>. Follow standard Microsoft Windows naming conventions. The limit is 127 characters, with no spaces allowed in the name.

File storage location. The file is saved in the location specified by the SAVEON:FILE:DEST command.

Default names. The default file name is SaveOnEvent. This name will be appended under certain circumstances. If you have previously selected (through the oscilloscope interface) to save channel 1 and channel 2 waveforms, for example, two files are saved, and the default file names will reflect the channel. If you have selected to save bus B1, the default file name will reflect that. The file name also indicates whether it is an image, measurement, or waveform.

Autoincrement. If SAVEON:FILE:AUTOInc is enabled, a number will be added to the file name and will be incremented for each new file. For example, if three events trigger the oscilloscope to save a measurement, three files will be created, which might be named “SaveOnEvent1Meas,” “SaveOnEvent2Meas,” and “SaveOnEvent3Meas.”

Extensions. Although these files have extensions (CSV for measurements and WFM for waveforms, for example), you do not need to enter the extension. It will be added automatically, depending on what you are saving. The query SAVEON:FILE:NAME? will return the file name without the extension.

Group Save On

Syntax
```
SAVEON:FILE:NAME <string>
SAVEON:FILE:NAME?
```

Related Commands SAVEON:FILE:AUTOInc, SAVEON:FILE:DEST, SAVEON:FILE:TYPE

Arguments <string> is the file name you want to use.

Examples SAVEON:FILE:NAME“MaskFailure” sets the name of the file to MaskFailure.

SAVEON:FILE:NAME? might return “:SAVEON:FILE:NAME MaskFailure5”, indicating the name you set for the oscilloscope to use, with the autoincrement number (5) appended.

SAVEON:FILE:TYPE

Sets or queries whether to use the data and time as the file name (auto) or to use a custom file name.

Group Save On

Syntax
```
SAVEON:FILE:TYPE {AUTO|CUSTOM}
SAVEON:FILE:TYPE?
```

Related Commands SAVEON:FILE:COUNt, SAVEON:FILE:NAME

Arguments AUTO uses the date and time as the file name for the saved events. An example auto file name is: 20110711_182946.

CUSTOM uses the file name that you specified using the SAVEON:FILE:NAME command.

Examples

`SAVEON:FILE:TYPE AUTO` sets the oscilloscope to use the file name and date (at the time of the save) as the file name when it saves an image, measurement, or waveform on a trigger.

`SAVEON:FILE:TYPE?` might return `:SAVEON:FILE:TYPE CUSTOM`, indicating that the oscilloscope will use the file name that you specified when it saves an image, measurement, or waveform on a trigger.

SAVEON:IMAGe

Sets or queries whether to save a screen capture when any of the following triggers occurs:

- Limit test failure – if set to On. (SAVEON:LIMit)
- Mask failure – if set to On. (SAVEON:MASK)
- Trigger – if set to On. (SAVEON:TRIGger)

The image will be saved to the file you selected with SAVEON:FILE:NAME, in the location that you selected using SAVEON:FILE:DEST. You can set options for file storage (such as file name, file destination, and autoincrement), using the SaveOn File commands. You can set the file type for the image to be stored in (such as BMP or JPG), through the oscilloscope interface (the SaveOn event setup menu).

Group

Save On

Syntax

```
SAVEON:IMAGe {<NR1>|OFF|ON}
SAVEON:IMAGe?
```

Related Commands

SAVEON:FILE:DEST, SAVEON:FILE:NAME, SAVEON:LIMit, SAVEON:MASK, SAVEON:TRIGger

Arguments

`<NR1>` = 0 disables Save On Image; any other value turns this feature on.

`OFF` disables Save On Image.

`ON` enables Save On Image.

Examples

`SAVEON:IMAGE ON` sets the oscilloscope to save a screen capture on a specified trigger.

`SAVEON:IMAGE?` might return `:SAVEON:IMAGE 1`, indicating that the oscilloscope will save a screen capture when the specified trigger occurs.

SAVEON:LIMit

Sets or queries whether to save a file when there is a limit test failure.

When this is set to On, a limit test failure will trigger the instrument to save an image, a measurement, and/or a waveform to a file, depending on what you specify using the Related Commands listed below. For example, if you set SAVEON:IMAGe to On, the instrument will save a screen capture to the file you selected with SAVEON:FILE:NAME, in the location that you selected using SAVEON:FILE:DEST. You can set options for file storage (such as file name, file destination, and autoincrement), using the SaveOn File commands.

Analog and math waveforms are saved using one file per waveform. Digital waveforms are all saved to a single file.

Group Save On

Syntax

```
SAVEON:LIMit {<NR1>|OFF|ON}
SAVEON:LIMit?
```

Related Commands

Arguments

<NR1> = 0 disables Save On Limit; any other value turns this feature on.

OFF disables Save On Limit.

ON enables Save On Limit.

Examples

SAVEON:LIMIT ON sets the oscilloscope to save (image, measurement, and/or waveform) files when there is a limit test failure.

SAVEON:LIMIT ? might return :SAVEON:LIMIT OFF, indicating that files will not be saved when there is a limit test failure.

SAVEON:MASK

Sets or queries whether to save a file when there is a mask failure.

When SAVEON:MASK is enabled, a mask failure will trigger the instrument to save an image, a measurement, and/or a waveform to a file, depending on what you specify using the Related Commands listed below. When there is a mask failure, if you have enabled SAVEON:IMAGe, the instrument will save a screen capture to the file you selected using SAVEON:FILE:NAME and SAVEON:FILE:DEST. You can set options for file storage (such as file name, file destination, and autoincrement), using the SaveOn File commands.

Use the oscilloscope interface to select whether to save one or more analog channels, digital channels, or math waveforms

Analog and math waveforms are saved using one file per waveform. Digital waveforms are all saved to a single file.

Group Save On

Syntax
```
SAVEON:MASK {<NR1>|OFF|ON}
SAVEON:MASK?
```

Related Commands SAVEON:FILE:DEST, SAVEON:FILE:NAME, SAVEON:IMAGe, SAVEON:MEASUrement, SAVEON:WAVEform

Arguments <NR1> = 0 disables Save On Mask; any other value turns this feature on.

OFF disables Save On Mask.

ON enables Save On Mask.

Examples SAVEON:MASK ON sets the oscilloscope to save (image, measurement, and/or waveform) files when there is a mask failure.

SAVEON:MASK ? might return :SAVEON:MASK OFF, indicating that files will not be saved on mask failure.

SAVEON:MEASUrement

Sets or queries whether to save a measurement when any of the following triggers occurs:

- Limit test failure – if set to On. (SAVEON:LIMit)
- Mask failure – if set to On. (SAVEON:MASK)
- Trigger – if set to On. (SAVEON:TRIGger)

The image will be saved to the file you selected with SAVEON:FILE:NAME, in the location that you selected using SAVEON:FILE:DEST. You can set options for file storage (such as file name, file destination, and autoincrement), using the SaveOn File commands.

Group Save On

Syntax

```
SAVEON:MEASUrement {<NR1>|OFF|ON}
SAVEON:MEASUrement?
```

Related Commands

SAVEON:FILE:DEST, SAVEON:FILE:NAME, SAVEON:LIMit, SAVEON:MASK, SAVEON:TRIGger

Arguments

<NR1> = 0 disables Save On Measurement; any other value turns this feature on.

OFF disables Save On Measurement.

ON enables Save On Measurement.

Examples

SAVEON:MEASUREMENT ON to save measurements when specified triggers occur.

SAVEON:MEASUREMENT ? might return :SAVEON:MEASUREMENT 0, indicating that a measurement will not be saved when triggers occur.

SAVEON:NUMEvents

Sets or queries the maximum number of events that will be saved. You can use this feature to avoid running out of disk space, especially if you save large files.

Group

Save On

Syntax

```
SAVEON:NUMEvents <NR3>
SAVEON:NUMEvents?
```

Related Commands

SAVEON:COUNt?,

Arguments

<NR3> specifies the number of events that will be saved before a reset is required.

Examples

SAVEON:NUMEVENTS 10 sets the number of events that will be saved to 10.

SAVEON:NUMEVENTS ? might return :SAVEON:NUMEvents 30, indicating that the next 30 events will be saved to files. After that, no files will be saved until you reset the counter.

SAVEON:SETUP

Saves the instrument setup when there is an event. Queries the status of this feature (on or off).

Group Save On

Syntax
```
SAVEON:SETUP {<NR1>|ON|OFF}
SAVEON:SETUP?
```

Arguments <NR1> = 0 disables the feature; any other value turns this feature on.

ON enables the feature.

OFF disables the feature.

Examples SAVEON:SETUP ON sets the oscilloscope to save the instrument setup when there is an event (limit test failure, mask failure, or trigger).

SAVEON:SETUP? might return :SAVEON:SETUP 0, indicating that the oscilloscope will not save the instrument setup when there is an event.

SAVEON:TRIGger

Sets or queries whether to save a file when a trigger occurs. You can define the trigger using Trigger commands or the oscilloscope interface.

The trigger will cause the instrument to save an image, a measurement, and/or a waveform to a file, depending on what you specified (see Related Commands below). For example, if you have set SAVEON:IMAGe to On, and a trigger event occurs, the instrument will save a screen capture. You can set options for file storage (such as file name, file destination, and autoincrement), using the SaveOn File commands.

Use the oscilloscope interface to select whether to save one or more analog channels, digital channels, or math waveforms

Analog and math waveforms are saved using one file per waveform. Digital waveforms are all saved to a single file.

Group Save On

Syntax
```
SAVEON:TRIGger {<NR1>|ON|OFF}
SAVEON:TRIGger?
```

Related Commands SAVEON:IMAGe, SAVEON:MEASUrement, SAVEON:WAVEform

Arguments <NR1> = 0 disables Save On Trigger; any other value turns this feature on.

OFF disables Save On Trigger.

ON enables Save On Trigger.

Examples

SAVEON:TRIGGER ON sets the oscilloscope to save an image, measurement, and/or waveform when a trigger occurs.

SAVEON:TRIGGER ? might return :SAVEON:TRIGGER ON, indicating that a file will be saved on triggering.

SAVEON:WAVEform

Sets or queries whether to save a waveform when any of the following triggers occurs:

- Limit test failure – if set to On. (SAVEON:LIMit)
- Mask failure – if set to On. (SAVEON:MASK)
- Trigger – if set to On. (SAVEON:TRIGger)

The waveform will be saved to the file you selected with SAVEON:FILE:NAME, in the location that you selected using SAVEON:FILE:DEST. You can set options for file storage (such as file name, file destination, and autoincrement), using the SaveOn File commands.

Group

Save On

Syntax

```
SAVEON:WAVEform {<NR1>|ON|OFF}
SAVEON:WAVEform?
```

Related Commands

SAVEON:FILE:DEST, SAVEON:FILE:NAME, SAVEON:LIMit, SAVEON:MASK, SAVEON:TRIGger

Arguments

<NR1> = 0 disables Save On Waveform; any other value turns this feature on.

OFF disables Save On Waveform.

ON enables Save On Waveform.

Examples

SAVEON:WAVEFORM ON turns on the Save On Waveform feature, so that a waveform will be saved when a selected trigger occurs.

SAVEON:WAVEFORM ? might return :SAVEON:WAVEFORM ON, indicating that a waveform will be saved when a selected trigger occurs.

*SDS (No Query Form)

This command (no query form) changes the specified setup to reference the factory setup instead of the specific user setup slot. The content of the setup slot is unchanged, but the data will no longer be accessible to you.

This command is equivalent to selecting Delete from the File menu, and then choosing the specified setup.

Group Save and Recall

Syntax `*SDS <NR1>`

Related Commands DELEte:SETUp

Arguments `<NR1>` specifies a user setup location to delete. Setup storage location values range from 1 through 10; using an out-of-range value causes an error.

Examples `*SDS2` changes setup slot 2 to reference the factory setup.

SEARCH:MARKALLevents

This command sets or queries the mark all trigger events setting.

Group Search and Mark

Syntax
```
SEARCH:MARKALLevents {<NR1>|OFF|ON}
SEARCH:MARKALLevents?
```

Arguments `<NR1>` = 0 disables the function; any other value enables it.

`OFF` disables the function.

`ON` enables the mark all trigger events function.

Examples `SEARCH:MARKALLEVENTS ON` turns on the mark all trigger events feature.

`SEARCH:MARKALLEVENTS?` might return `:SEARCH:MARKALLEVENTS 0`, indicating that mark all trigger events is turned off.

SEARCH:SEARCH<x>

This command converts the specified search marks to the user marks. <x> can be 1 to 8.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x> <QString>
SEARCH:SEARCH<x>?
```

Arguments <QString> specifies the user search marks.

Examples SEARCH:SEARCH1 TOUSER converts the Search 1 mark to the user marks.

SEARCH:SEARCH1? might return :SEARCH:SEARCH1 TOUSER, indicating that all the search1 marks are converted to user marks.

SEARCH:SEARCH<x>:COPy (No Query Form)

This command (no query form) copies the search criteria to the trigger, the trigger criteria to a search, or copy search to search. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:COPy
{SEARCHtotrigger|TRIGgertosearch|FROMSearch<x>}
```

Arguments SEARCHtotrigger copies the search criteria to the trigger.

TRIGgertosearch copies the trigger criteria to the search.

FROMSearch<x> copies the search criteria to the search, where <x> is 1 to 8.

Examples SEARCH:SEARCH1:COPY TRIGGERTOSEARCH copies the trigger criteria to the search 1 criteria.

SEARCH:SEARCH<x>:STATE

This command sets the search state to on or off. <x> is the search number, which can be 1 to 8. The query form returns the search state.

Group Search and Mark

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.

Examples SEARCH:SEARCH2:STATE ON sets the search state for search 2 on.

SEARCH:SEARCH1:STATE? might return :SEARCH:SEARCH1:STATE 0, indicating that the search 1 is off.

SEARCH:SEARCH<x>:TOTAL? (Query Only)

This query-only command returns the total number of matches for the search. The total number of matches can be more than the number of marks placed. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TOTAL?
```

Examples SEARCH:SEARCH1:TOTAL? might return :SEARCH:SEARCH1:TOTAL 7, indicating that there are 7 matches for search 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:CONDition

This command sets or queries the CAN bus trigger condition. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:CONDition
{ERRor|DATA|IDANDDATA|EOF|IDentifier|ACKMISS|SOF|FRAMEtype}
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:CONDition?
```

Arguments Arguments specify the CAN bus trigger condition.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:CAN:CONDition EOF` sets the CAN bus trigger condition to end of frame.

`SEARCH:SEARCH1:TRIGger:A:BUS:CAN:CONDition?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:CONDITION SOF`, indicating the CAN bus trigger condition is set to start of frame.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:FORMat

This command sets or queries the CAN bus trigger data format. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:FORMat?
```

Arguments `BINary` specifies the pattern format as binary.

`HEX` specifies the pattern format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:FORMat HEX` sets the data format to HEX.

`SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:FORMat?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:DATA:FORMAT BINARY`, indicates that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:QUALifier

This command sets or queries the CAN bus trigger data qualifier. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:QUALifier
{EQUal|LESSEQual|MOREEQua|UNEQual|LESSthan|MOREthan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:QUALifier?
```

Arguments Arguments are the data qualifier types.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:QUALifier UNEQual` sets the data qualifier to unequal.

`SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:QUALifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:DATA:QUALIFIER EQUAL`, indicating that the data qualifier is set to equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:SIZe

This command sets or queries the CAN bus trigger data size in bytes. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:SIZe <nr3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:SIZe?
```

Arguments `<nr3>` specifies the data size.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:SIZe 1` sets the data size to 1.

`SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:SIZe?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:DATA:SIZE 1`, indicating the data size is set to 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:VALue

This command sets or queries the CAN bus trigger data value. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:VALue?
```

Arguments <string> is the data value.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:VALue "1010" sets the data value to XXXXXX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:CAN:DATa:VALue? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:DATA:VALUE "XXXXXXXXXX", indicating the data value is a don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:FRAMEtype

This command sets or queries CAN bus trigger frame type. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:FRAMEtype
{DATa|ERRor|OVERLoad|REMote}
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:FRAMEtype?
```

Arguments Arguments are the available frame types.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:CAN:FRAMEtype ERROR sets the frame type to error.

SEARCH:SEARCH1:TRIGger:A:BUS:CAN:FRAMEtype? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:FRAMETYPE DATA, indicates the frame type is set to data.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:DIRection

This command sets or queries the CAN bus identifier direction. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:DIRection
{DONTCare|READ|WRITE}
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:DIRection?
```

Arguments `DONTCare` specifies the direction is a don't care.

`READ` specifies the read direction.

`WRITE` specifies the write direction.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:DIRection`

`SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:DIRection?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:IDENTIFIER:DIRECTION DONTCARE`, indicating the direction is a don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:FORMat

This command sets or queries the CAN bus trigger identifier format. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:FORMat?
```

Arguments `BINary` specifies the identifier format as binary.

`HEX` specifies the identifier format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:FORMat HEX` sets the identifier format to hexadecimal.

SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:FORMat? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:IDENTIFIER:FORMAT BINARY, indicating that the identifier format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:MODe

This command sets or queries the CAN bus trigger identifier mode. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:MODe
{EXTENDed|STandard}
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:MODe?
```

Arguments EXTENDed specifies the extended identifier mode.

STandard specifies the standard identifier mode.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:MODe EXTEND sets the identifier mode to extended.

SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:MODe? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:IDENTIFIER:MODE STANDARD, indicating the identifier mode is set to standard.

SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:VALue

This command sets or queries CAN bus trigger identifier value. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:VALue?
```

Arguments <string> is the identifier value.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:VALue "1010" sets the identifier value to XXXXXX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:CAN:IDentifier:VALue? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:CAN:IDENTIFIER:VALUE "XXXXXXXXXXX", indicating the identifier value is a don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

This command specifies a field or condition within an Ethernet frame to search on.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition
{STARTFRAME|MACADDRess|MACTYPE|IPV4Header|TCPHeader|DATa
|ENDPACKET|FCSError|QTAG|TCPDATA}
```

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition?
```

Related Commands Most of the other TRIGger:A:BUS:B<x>:ETHERnet commands are impacted by the setting of this command.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier

Arguments

SFD – Start of frame delimiter.

MACADDRess – MAC addresses field.

MACLENgth – MAC length/type field.

IPHeader – IP header field. .

TCPHeader – TCP header field.

DATa – TCP/IPv4 or MAC protocol client data field.

EOP – End of Packet field.

IDLe – Idle field.

FCSError – Frame Check Sequence Error (CRC) field.

QTAG – IEEE 802.1Q (VLAN) control information field.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:CONDITION MACADDRESS specifies MACADDRess as the field within an Ethernet frame to search on.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:CONDITION? might return DATA, indicating that DATA is the currently specified field within an Ethernet frame to search on.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet data field. The search condition needs to be set to DATa.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:FORMat?
```

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:ETHERnet:DATa:FORMat HEX specifies HEX as the data format to use when searching on the Ethernet data field.

SEARCH:SEARCH1:TRIGger:A:BUS:ETHERnet:DATa:FORMat?might return BINARY, indicating that binary is the currently specified data format for searching on the Ethernet data field.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:OFFSet

This command specifies the data offset value, in bytes, to use when searching on the Ethernet data field. The search condition needs to be set to DATa.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:OFFSet <NR1>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:OFFSet?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:VALue

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe

Arguments

<NR1> is an integer whose minimum and default values are -1 (don't care) and maximum is 1,499.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:DATA:OFFSEt 36 sets the data offset to 36 bytes.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:DATA:OFFSET? might return -1, indicating that the data offset value is the default value, -1, meaning “don’t care”.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe

This command specifies the number of contiguous TCP/IPv4/MAC client bytes to use when searching on the Ethernet data field. The default is all X’s (don’t care). The search condition needs to be set to DATa.

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe <NR1>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier

Arguments

The minimum and default values are 1, and the maximum value is 16.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:DATA:SIZE 4 specifies 4 as the number of contiguous TCP/IPv4/MAC client bytes to use when searching on the Ethernet data field.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:DATA:SIZE? might return 6, indicating that 6 is the currently specified number of contiguous TCP/IPv4/MAC client bytes to use when searching on the Ethernet data field.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:VALue

This command specifies the binary value to use when searching on the Ethernet data field. The default is all X's (don't care). The search condition needs to be set to DATa.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:OFFSet

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe

Arguments QString is a quoted string where the allowable characters are 0, 1, and X. The allowable number of characters depends on the setting for size (using SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe). The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:DATA:VALUE "00001000" specifies 00001000 as the value to use when triggering on the Ethernet binary data field, assuming that TRIGGER:A:BUS:ETHERNET:DATA:SIZe is set to 1 byte.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:DATA:VALUE? might return "00001000", indicating 0000100 is the currently specified value used when searching on the Ethernet binary data field.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet IPv4 header address destination field. The search condition needs to be set to IPHeader.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:
DESTinationaddr:FORMat {BINary|HEX|DECImal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:
DESTinationaddr:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

`DECImal` specifies the data format as decimal.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:FORMAT HEX` specifies `HEX` as the data format to use when searching on the Ethernet IPv4 header address destination field.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:FORMAT?` might return `BINARY`, indicating binary is the currently specified data format used when triggering on the Ethernet IPv4 header address destination field.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue

This command specifies the 32-bit value to use when searching on the Ethernet IPv4 header address destination field. The default is all X's (don't care). The search condition needs to be set to `IPHeader`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:
DESTinationaddr:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:
DESTinationaddr:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat

Arguments QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:VALUE "11000000101010000000000000000001"` specifies 192.168.0.1 as the value to use when searching on the Ethernet IPv4 header address destination field.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:VALUE?` might return `"11000000101010000000000000000001"`, indicating that 192.168.0.1 is the currently specified value used when searching on the Ethernet IPv4 header address destination field.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat

This command specifies the data format (binary or hex) of the bit pattern to use when searching on the Ethernet IPv4 header protocol field. The search condition needs to be set to `MACADDRess`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:FORMAT HEX specifies HEX as the data format to use when searching on the Ethernet IPv4 header protocol field.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:FORMAT? might return BINARY, indicating that binary is the currently specified data format used when searching on the Ethernet IPv4 header protocol field.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue

This command specifies the 8-bit value to use when searching on the Ethernet IPv4 header protocol field. The default is all X's (don't care). The search condition needs to be set to IPHeader.

NOTE. *Commonly used protocol values are 1 (ICMP), 2 (IGMP), 6 (TCP) and 17 (UDP).*

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:
VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:
VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat

Arguments

QString is a quoted string of up to 8 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:VALUE "01010010" specifies 01010010 as the value to use when searching on the Ethernet IPv4 header protocol field.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:VALUE? might return "XXXXXXXX", indicating that the trigger value has been set to "don't care".

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat

This command specifies the data format (binary or hex) of the bit pattern to use when searching on the Ethernet IPv4 header address source field. The search condition needs to be set to `IP header`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:
FORMat {BINary|HEX|DECImal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:
FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

`DECImal` specifies the data format as decimal.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:FORMAT HEX` specifies to use the `HEX` data format when searching on the Ethernet IPv4 header address source field.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:FORMAT?` might return `BINARY`, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue

This command specifies the 32-bit value to use when searching on the Ethernet IPv4 header address source field. The default is all X's (don't care). The search condition needs to be set to `IPHeader`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:
VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:
VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat

Arguments

QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:VALUE "11000000101010000000000000000001"` specifies to use the value of 192.168.0.1 when searching on the Ethernet IPv4 header address source field.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:VALUE` might return `"11000000101010000000000000000001"`, indicating that the search value has been set to 192.168.0.1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat

This command specifies the data format (binary or hex) of the bit pattern to use when searching on the Ethernet MAC address destination field. The search condition needs to be set to `MACADDRess`.

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:
DESTination:FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:
DESTination:FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:FORMAT HEX specifies to use the HEX data format when searching on the Ethernet MAC address destination field.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:FORMAT might return BINARY, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue

This command specifies the 48-bit value to use when searching on the Ethernet MAC address destination field. The default is all X's (don't care). The search condition needs to be set to MACADDRess.

NOTE. *MAC Addresses are 48-bit values such as 08:00:11:1E:C9:AE hex.*

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:
DESTination:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:
DESTination:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat

Arguments

QString is a quoted string of up to 48 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:VALUE "XXXXXXXX00110101111111000000011110101010011001000" specifies to use the value of XX:35:FC:07:AA:C8 hex when searching on the Ethernet MAC address destination field.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:VALUE? might return "XXXXXXXX00110101111111000000011110101010011001000", indicating a MAC address destination field value of XX:35:FC:07:AA:C8 hex.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet MAC address source field. The search condition needs to be set to MACADDRess.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:
FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:
FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:FORMAT HEX specifies to use the HEX data format when searching on the Ethernet MAC address source field.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:FORMAT? might return BINARY, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue

This command specifies the 48-bit value to use when searching on the Ethernet MAC address source field. The default is all X's (don't care). The search condition needs to be set to MACADDRess.

NOTE. *MAC Addresses are 48-bit values such as 08:00:11:1E:C9:AE hex.*

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:
VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:
VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat

Arguments QString is a quoted string of up to 48 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:VALUE "XXXXXXXX001101011111110000000111101010101100100 0"` specifies to use the value of XX:35:FC:07:AA:C8 hex when searching on the Ethernet MAC address source field.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:VALUE?` might return `"XXXXXXXX00110101111111000000011110101010110010 00"`, indicating a MAC address source field value of XX:35:FC:07:AA:C8 hex.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet MAC length or type. The search condition needs to be set to `MACADDRess`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:
FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:
FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:TYPE:FORMAT HEX specifies to use the HEX data format when searching on the Ethernet MAC type.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:TYPE:FORMAT? might return BINARY, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue

This command specifies the 16-bit value to use when searching on the Ethernet MAC length/type. The default is all X's (don't care). The search condition needs to be set to MACADDRess.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:
VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:
VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat

Arguments QString is a quoted string of up to 16 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:TYPE:VALUE "XXXXXXXX00001000" specifies to use the hexadecimal value XX08 when searching on the Ethernet MAC type.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:MAC:TYPE:VALUE? might return "XXXXXXXX00001000", indicating an Ethernet MAC type value of XX08 hex.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:FORMat

This command specifies the data format (binary or hex) to use when searching on the Q-Tag field. The search condition needs to be set to QTAG.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:QTAG:FORMAT HEX specifies to use the HEX data format when searching on the Ethernet Q-Tag field.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:QTAG:FORMAT? might return BINARY, indicating that the data format to use for searching on the Ethernet Q-Tag field has been set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:VALue

This command specifies the 32-bit value to use when searching on the Ethernet Q-Tag field. The default is all X's (don't care). The search condition needs to be set to QTAG.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:FORMat

Arguments

QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:QTAG:VALUE` "XXXXXXXXXXXXXXXXXXXX010010001010" specifies to use the value of hexadecimal XXXXX48A when searching on the Ethernet Q-Tag field.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:QTAG:VALUE?` might return "XXXXXXXXXXXXXXXXXXXX010010001010", indicating that hexadecimal XXXXX48A has been set as the Ethernet Q-Tag field search value.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier

This command specifies the qualifier to be used when the Ethernet search condition is set to `MACLENgth` or `DATa`. Normally, the Ethernet qualifier is set to "equal to".

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier
{LESSthan|MOREthan|EQual|UNEQual|LESSEQual|MOREEQual}

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

Arguments

`LESSthan` sets the qualifier to less than.

`MOREthan` sets the qualifier to greater than.

`EQual` sets the qualifier to equal.

`UNEQual` sets the qualifier to not equal.

`LESSEQual` sets the qualifier to less than or equal.

MOREEQual sets the qualifier to greater than or equal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:QUALIFIER LESSTHAN sets the qualifier to “less than”.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header acknowledgement number. The search condition needs to be set to TCPHeader.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:
FORMat {BINary|HEX|DECImal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:
FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDitionSEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

DECImal specifies the data format as decimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:FORMAT HEX specifies to use the HEX data format when searching on the Ethernet TCP header acknowledgement number.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:FORMAT? might return BINARY, indicating that the data format to use when triggering on the Ethernet TCP header acknowledgement number has been set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue

This command specifies the 32-bit value to use when searching on the Ethernet TCP header acknowledgement number. The default is all X’s (don’t care). The search condition needs to be set to TCPHeader.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:
VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:
VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat

Arguments QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:VALUE` "XXXXXXXXXXXXXXXXXXXX00001000" specifies hexadecimal XXXXXX08 as the value to use when searching on the Ethernet TCP header acknowledgement number.

`TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:VALUE?` might return "XXXXXXXXXXXXXXXXXXXX00001000", indicating that hexadecimal XXXXXX08 has been specified as the value to use when searching on the Ethernet TCP header acknowledgement number.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header destination port number. The search condition needs to be set to `TCPHeader`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
DESTinationport:FORMat {BINary|HEX|DECImal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
DESTinationport:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

DECImal specifies the data format as decimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:FORM HEX specifies to use the HEX data format when searching on the Ethernet TCP header destination port number.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:FORM might return BINARY, indicating that the data format to use when searching on the Ethernet TCP header destination port number has been set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue

This command specifies the 16-bit value to use when searching on the Ethernet TCP header destination port number. The default is all X's (don't care). The search condition needs to be set to TCPHeader.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
DESTinationport:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
DESTinationport:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat

Arguments QString is a quoted string of up to 16 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:VALUE "XXXXXXXX00100010" specifies to use the value of hexadecimal XX22 when searching on the Ethernet TCP header destination port number.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:VALUE? might return "XXXXXXXXXXXXXXXX", indicating that hexadecimal XX22 has been set as the value to use when searching on the Ethernet TCP header destination port number.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header sequence number. The search condition needs to be set to TCPHeader.

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:
FORMat {BINary|HEX|DECImal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:
FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

DECImal specifies the data format as decimal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:FORMAT HEX specifies to use the HEX data format when searching on the Ethernet TCP header sequence number.

SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:FORMAT? might return BINARY, indicating that the binary data format has been specified to use when searching on the Ethernet TCP header sequence number.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue

This command specifies the 32-bit value to use when searching on the Ethernet TCP header sequence number. The default is all X's (don't care). The search condition needs to be set to `TCPHeader`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:
VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:
VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat

Arguments QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:VALUE "XXXXXXXXXXXXXXXXXXXX000100010001"` specifies to use the value of hexadecimal XXXXX111 when searching on the Ethernet TCP header sequence number.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:VALUE?` might return "XXXXXXXXXXXXXXXXXXXXXX0010010100" , indicating that hexadecimal XXXXX111 has been specified as the value to use when searching on the Ethernet TCP header sequence number. .

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the Ethernet TCP header source port number. The search condition needs to be set to `TCPHeader`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
SOUrceport:FORMat {BINary|HEX|DECImal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
SOUrceport:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

`DECImal` specifies the data format as decimal.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:FORMAT HEX` specifies to use the `HEX` data format when searching on the Ethernet TCP header source port number.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:FORMAT?` might return `BINARY`, indicating that the binary data format has been specified to use when searching on the Ethernet TCP header source port number.

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue

This command specifies the 16-bit value to use when searching on the Ethernet TCP header source port number. The default is all X's (don't care). The search condition needs to be set to `TCPHeader`.

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
SOUrceport:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:
SOUrceport:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat

Arguments QString is a quoted string of up to 16 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:VALUE "XXXX000010100110"` specifies to use the value of hexadecimal X0A6 when searching on the Ethernet TCP header source port number.

`SEARCH:SEARCH1:TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:VALUE?` might return "XXXXX01001010110", indicating that hexadecimal X0A6 has been specified as the value to use when searching on the Ethernet TCP header source port number.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CONDition

This command sets or queries FLEXRAY bus search trigger condition. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CONDition
{SOF|FRAMEtype|IDentifier|CYCLEcount|HEADer|DATA|IDANDDATA|EOF|ERRor
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CONDition?
```

Arguments Arguments are the available trigger conditions.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CONDition SOF` sets the trigger condition to start of frame.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CONDition?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:CONDITION SOF`, indicating the FLEXRAY trigger condition is start of frame.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat

This command sets or queries the FLEXRAY bus search cycle count format. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat?
```

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat HEX` sets the cycle count format to hexadecimal.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:CYCLECOUNT:FORMAT BINARY`, indicating the cycle count format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier

This command sets or queries the FLEXRAY bus search trigger cycle count qualifier. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier?
```

Arguments Arguments are the available cycle count qualifiers.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier LESSthan` sets the cycle count qualifier to less than.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:CYCLECOUNT:QUALIFIER EQUAL`, indicating the cycle count qualifier is set to equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue

This command sets or queries the FLEXRAY bus search cycle count value. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue?
```

Arguments `<string>` specifies the cycle count value.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue “1010”` sets the cycle count value to XX1010.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:CYCLECOUNT:VALUE "XXXXXX"`, indicating the cycle count value is don’t care, and it will trigger on any cycle count.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:FORMat

This command sets or queries FLEXRAY bus search data format. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:FORMat?
```

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:FORMat HEX` sets the data format to hexadecimal.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:FORMat?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:DATA:FORMAT BINARY`, indicating the data format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet

This command sets or queries the FLEXRAY bus search trigger data offset. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet <nr3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet?
```

Arguments `<nr3>` specifies the data offset in bytes. A setting of X/Don't care is accomplished by setting the offset to –1.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet 2` sets the data offset to 2 bytes.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:DATA:OFFSET -1`, indicating the data offset is don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier

This command sets or queries the FLEXRAY bus search trigger data qualifier. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier?
```

Arguments Arguments are the available data qualifiers.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier LESSTHAN` sets the data qualifier to less than.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:DATA:QUALIFIER EQUAL`, indicating the data qualifier is equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:SIZe

This command sets or queries the FLEXRAY bus search trigger data size in bytes. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:SIZe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:SIZe?
```

Arguments `<NR3>` specifies the data size in bytes. A data size of –1 is don't care.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:SIZe 2` sets the data size to 2.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:SIZe?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:DATA:SIZE 1`, indicating the data size is 1 byte.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:VALue

This command sets or queries the FLEXRAY bus search trigger data value. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:VALue <STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:VALue?
```

Arguments

<STRING> specifies the data value.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:VALue "1010" sets the data value to XXXXXX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:DATa:VALue? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:DATA:VALUE "XXXXXXXXXX", indicating the data value is a don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:EOFTYPE

This command sets or queries the FLEXRAY bus search trigger end of file type. The search number is specified by x.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:EOFTYPE
{ALL|STATIC|DYNAMIC}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:EOFTYPE?
```

Arguments

ALL specifies either end of file type.

STATIC specifies the static end of file type.

DYNAMIC specifies the static end of file type.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:EOFTYPE STATIC sets the end of file type to static.

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:EOFTYPE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:EOFTYPE ALL, indicating either end of file type will cause a trigger.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:ERRTYPE

This command sets or queries the FLEXRAY bus search trigger error type. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:ERRTYPE
{CRCHEADer|CRCTRAILer|NULLFRSTATIC|NULLFRDYNAMIC
|SYNCFRAME|STARTUPNOSYNC}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:ERRTYPE?
```

Arguments Arguments are the available error types.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:ERRTYPE SYNCFRAME` sets the error type to SYNCFRAME.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:ERRTYPE?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:ERRTYPE CRCHEADER`, indicating the error type is CRCHEADER.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:FRAMEType

This command sets or queries the FLEXRAY bus search trigger frame type. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:FRAMEType
{NORMal|PAYload|NULL|SYNC|STARTup}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:FRAMEType?
```

Arguments Arguments are the available frame types.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:FRAMEType startup` sets the frame type to startup.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:FRAMEType?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:FRAMETYPE NORMAL`, indicating the frame type is normal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC

This command sets or queries the FLEXRAY bus search trigger header CRC. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC <STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC?
```

Arguments <STRING> specifies the CRC.

Examples SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC "1010" sets the header CRC to XXXXXX101.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:HEADER:CRC "XXXXXXXXXX", indicating the header CRC is a don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount

This command sets or queries the FLEXRAY bus search trigger header cycle count. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount
<STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount?
```

Arguments <STRING> specifies the header cycle count.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount "1010" sets the header cycle count to XX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:HEADER:CYCLECOUNT "XXXXXX" indicating the cycle count has not been set.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID

This command sets or queries the FLEXRAY bus search trigger header frame id. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID
<STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID?
```

Arguments <STRING> specifies the frame ID.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID “1010” sets the header frame ID to XXXXXXX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:HEADER:FRAMEID "XXXXXXXXXXX", indicating the frame ID is a don’t care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits

This command sets or queries the FLEXRAY bus search header Indicator Bits. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits
<STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits?
```

Arguments <STRING> specifies the header Indicator Bits.

Examples SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits “1010” sets the header Indicator Bits to X1010.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:HEADER:INDBITS "XXXXX", indicating the Indicator Bits have not been set.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth

This command sets or queries the FLEXRAY bus search trigger header payload length. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth
<STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth?
```

Arguments <STRING> specifies the header Payload Length.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth “1010” sets the Payload Length to 1010.

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:HEADER:PAYLENGTH "XXXXXXX", indicating the Payload Length has not been set.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat

This command sets or queries the FLEXRAY bus search trigger identifier format. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat?
```

Arguments `BINary` specifies the identifier format as binary.

`HEX` specifies the identifier format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat HEX` sets the identifier format to hexadecimal.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:IDENTIFIER:FORMAT BINARY`, indicating the identifier format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier

This command sets or queries the FLEXRAY bus search trigger identifier qualifier. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier?
```

Arguments Arguments are the available identifier qualifiers.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier LESSTHAN` sets the identifier qualifier to less than.

`SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:IDENTIFIER:QUALIFIER EQUAL`, indicating the identifier qualifier is equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue

This command sets or queries the FLEXRAY bus search trigger identifier value. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue
<STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue?
```

Arguments <STRING> specifies the identifier value.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue “1010” sets the identifier value to XXXXXXX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:FLEXRAY:IDENTIFIER:VALUE "XXXXXXXXXX", indicating the identifier value is a don’t care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:FORMat

This command sets or queries the address format for the I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:FORMat?
```

Arguments BINary specifies the address format as Binary.

HEXadecimal specifies the address format as Hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:FORMAT HEX sets the address format for I2C bus trigger search 1 as hexadecimal.

SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:FORMAT? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:FORMAT BINARY, indicating that the address format for I2C bus trigger search 1 is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:MODe

This command sets or queries the address mode for the specified I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:MODe
{ADDR10|ADDR7}
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:MODe?
```

Arguments `ADDR10` specifies the address mode as ADDR10.

`ADDR7` specifies the address mode as ADDR7.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:MODE ADDR10` sets the address mode for I2C bus trigger search 1 to ADDR10.

`SEARCH:SEARCH2:TRIGGER:A:BUS:I2C:ADDRESS:MODE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:I2C:ADDRESS:MODE ADDR7`, indicating that the address mode for I2C bus trigger search 2 is set to ADDR7.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:TYPe

This command sets or queries the address type for the specified I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:TYPe
{CBUS|EEPROm|GENeralcall|HSMODe|NONe|STARtbyte}
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:TYPe?
```

Arguments `CBUS` specifies the address type as CBUS.

`EEPROm` specifies the address type as EEPROM.

`GENeralcall` specifies the address type as General Call.

HSMODe specifies the address type as High Speed Mode.

NONe specifies the address type as None.

STARtbyte specifies the address type as Start Byte.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:TYPE CBUS sets the address type for I2C bus trigger search 1 to CBUS.

SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:TYPE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:TYPE NONE, indicating that the address type for I2C bus trigger search 1 is set to NONe.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:VALue

This command sets or queries the address string when the search condition for the specified search is Address or AddressData.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:MODe, SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:FORMat

Arguments

<string> specifies the address value. This is either a 7-bit or 10-bit value depending on the address mode. The valid characters are 0-9, A-F, and X for addresses in hexadecimal format; and 0, 1, and X otherwise.

Examples

SEARCH:SEARCH2:TRIGGER:A:BUS:I2C:ADDRESS:VALUE "01XXXXX" sets the address value to "01XXXXX" when the mode is ADDR7 and the format is binary.

SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:ADDRESS:VALUE "XX", indicating that the address value is "XX" when the address mode is set to ADDR7 and the address format is hexadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:CONDition

This command sets or queries the trigger condition for the specified I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:CONDition
{ADDress|ADDRANDDATA|DATa|ACKMISS|REPEATstart|STARt|STOP}
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:CONDition?
```

Arguments

`ADDress` specifies the trigger condition as Address.

`ADDRANDDATA` specifies the trigger condition as Address and Data.

`DATa` specifies the trigger condition as Data.

`ACKMISS` specifies the trigger condition as Missing of Acknowledgement.

`REPEATstart` specifies the trigger condition as Repeat of Start.

`STARt` specifies the trigger condition as Start.

`STOP` specifies the trigger condition as Stop.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:CONDITION ADDRESS` sets the trigger condition for I2C bus trigger search 1 to Address.

`SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:CONDITION?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:CONDITION ADDRANDDATA`, indicating that the trigger condition for I2C bus trigger search 1 is set to Address and Data.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:DIRection

This command sets or queries the direction of the data for the I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:DIRection
{DONTCare|READ|WRITE}
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:DIRection?
```

Arguments

`DONTCare` specifies the direction of data as Don't Care.

`READ` specifies the direction of data as Read.

`WRITE` specifies the direction of data as Write.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:DIRECTION READ` sets the data direction for I2C bus trigger search 1 to READ.

`SEARCH:SEARCH2:TRIGGER:A:BUS:I2C:DATA:DIRECTION?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:I2C:DATA:DIRECTION DONTCARE`, indicating that the data direction of the I2C bus trigger search 2 is DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:FORMat

This command sets or queries the trigger data format for the specified I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:FORMat?
```

Arguments

`BINary` specifies the trigger data format as Binary.

`HEX` specifies the trigger data format as Hexadecimal.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:FORMAT HEX` sets the trigger data format for I2C bus trigger search 1 to hexadecimal.

`SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:FORMAT?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:FORMAT BINARY`, indicating that the trigger data format for I2C bus trigger search 1 is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:SIZe

This command sets or queries the length of the data string in bytes used for the specified I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:SIZe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:SIZe?
```

Arguments <NR3> specifies the data size in bytes.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:SIZE 1` sets the length of the data string for I2C bus trigger search 1 to 1 byte.

`SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:SIZE?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:SIZE 3`, indicating that the length of the data string for I2C bus trigger search 1 is 3 bytes.

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:VALue

This command sets or queries the data value of the data token for the specified I2C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:VALue?
```

Arguments <string> specifies the data value. The valid characters are 0, 1, or X for binary format; and A-F, 0-9, and X for hexadecimal format.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:I2C:DATA:VALUE "1001"` sets the data value for I2C bus trigger search 1 to "1001".

SEARCH:SEARCH2:TRIGGER:A:BUS:I2C:DATA:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:I2C:DATA:VALUE "XX", indicating that the data value for I2C bus trigger search is "XX" in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:CONDition

This command sets or queries the LIN bus search trigger condition. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:CONDition
{DATA|IDANDDATA|ERROR|IDentifier|SLEEP|SYNC|WAKEup}
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:CONDition?
```

Arguments Arguments are the available trigger conditions.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:LIN:CONDition DATA sets the trigger condition to data.

SEARCH:SEARCH1:TRIGger:A:BUS:LIN:CONDition? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:LIN:CONDITION SYNC, indicating the trigger condition is sync.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:FORMat

This command sets or queries the LIN bus search trigger data format. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:FORMat?
```

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples `SEARCH:SEARCH2:TRIGger:A:BUS:LIN:DATa:FORMat HEX` sets the data format to hexadecimal.

`SEARCH:SEARCH1:TRIGger:A:BUS:LIN:DATa:FORMat?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:LIN:DATA:FORMAT BINARY`, indicating the data format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:QUALifier

This command sets or queries the LIN bus search trigger data qualifier. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:QUALifier?
```

Arguments Arguments are the available data qualifiers.

Examples `SEARCH:SEARCH2:TRIGger:A:BUS:LIN:DATa:QUALifier LESS` sets the data qualifier to less than.

`SEARCH:SEARCH1:TRIGger:A:BUS:LIN:DATa:QUALifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:LIN:DATA:QUALIFIER EQUAL`, indicating the data qualifier is equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:SIZe

This command sets or queries the LIN bus search trigger data size in bytes. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:SIZe <nr3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:SIZe?
```

Arguments

<nr3> specifies the data size.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:LIN:DATa:SIZe 1.0 sets the data size to 1.

SEARCH:SEARCH1:TRIGger:A:BUS:LIN:DATa:SIZe? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:LIN:DATA:SIZE 1, indicating the data size is 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:VALue

This command sets or queries the LIN bus search trigger data value. The search number is specified by x.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:VALue?
```

Arguments

<string> specifies the data value.

Examples

SEARCH:SEARCH2:TRIGger:A:BUS:LIN:DATa:VALue 1010 sets the data value to XXXXXX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:LIN:DATa:VALue? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:LIN:DATA:VALUE "XXXXXXXXXX", indicating that the data value is a don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:ERRTYPE

This command sets or queries the LIN bus search trigger error type. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:ERRTYPE
{CHecksum|PARity|SYNC}
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:ERRTYPE?
```

Arguments `CHecksum` specifies the error type is checksum.

`PARity` specifies the error type is parity.

`SYNC` specifies the error type is sync.

Examples `SEARCH:SEARCH2:TRIGger:A:BUS:LIN:ERRTYPE Parity` sets the error type to parity.

`SEARCH:SEARCH2:TRIGger:A:BUS:LIN:ERRTYPE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:LIN:ERRTYPE SYNC`, indicating that the error type is sync.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:FORMat

This command sets or queries the LIN bus search trigger identifier format. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:FORMat?
```

Arguments `BINary` specifies the identifier format as binary.

`HEX` specifies the identifier format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:LIN:IDentifier:FORMat BINARY` sets the identifier format to binary.

`SEARCH:SEARCH1:TRIGger:A:BUS:LIN:IDentifier:FORMat?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:LIN:IDENTIFIER:FORMAT BINARY`, indicating that the identifier format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:VALue

This command sets or queries the LIN bus search identifier value. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:VALue?
```

Arguments <string> specifies the identifier value.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:LIN:IDentifier:VALue 1010 sets the identifier value to XX1010.

SEARCH:SEARCH1:TRIGger:A:BUS:LIN:IDentifier:VALue? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:LIN:IDENTIFIER:VALUE "XXXXXX", indicates the identifier value is don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat

This command sets the data format of the address (either binary or hex) to use in a MIL-STD-1553 bus command word search. The search condition needs to be set to COMMAND.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:
FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:
FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples

SEARCH:SEARCH<X>:TRIGGER:A:BUS:MIL1553B:COMMAND:ADDRESS:FORMAT HEX sets the data format to HEX.

SEARCH:SEARCH<X>:TRIGGER:A:BUS:MIL1553B:COMMAND:ADDRESS:FORMAT? might return BINARY, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier

This command specifies the qualifier to use when searching on a MIL-STD-1553 bus command word remote terminal address field. The search condition needs to be set to COMMAND.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:
ADDRess:QUALifier {LESSthan|MOREthan|EQual
|UNEQual|LESSEQual|MOREEQual}
```

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:
QUALifier?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue

Arguments

LESSthan sets the Command Address qualifier to less than.

MOREthan sets the Command Address qualifier to greater than.

EQual sets the Command Address qualifier to equal.

UNEQual sets the Command Address qualifier to not equal.

LESSEQual sets the Command Address qualifier to less than or equal.

MOREEQual sets the Command Address qualifier to greater than or equal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:COMMAND:ADDRESS:QUALIFIER EQUAL sets the qualifier to be used with the remote terminal address to equal.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:COMMAND:ADDRESS:QUALIFIER? might return EQUAL.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue

This command specifies the value of the 5-bit remote terminal address to use in a MIL-STD-1553 bus command word search. The search condition needs to be set to COMMAND.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue <QString>
```

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier

Arguments QString is a quoted string of up to 5 characters, where the allowable characters are 0, 1, and X.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:ADDRESS:VALUE "01000" sets the remote terminal address to be used in the search to 01000.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:ADDRESS:VALUE? might return 01000.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt

This command specifies the bit pattern for the 5-bit Word Count/Mode Code sub-address field to use in a MIL-STD-1553 bus command word trigger. (Use the commandSEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress to specify Word Count or Mode Code.) In Word Count mode, this field defines the number of data words to be transmitted or received, depending on the T/R bit setting. (Use the commandSEARCH:SEARCH<x>:TRIGger:A:

BUS:MIL1553B:COMMAND:TRBit to set the T/R bit.) A word count value of 0 actually indicates a transfer of 32 data words.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt
<QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:TRBit

Arguments `QString` is a quoted string of up to 5 characters, where the allowable characters are 0, 1 and X.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:COMMAND:COUNT "01000"` sets the bit pattern for the Word Count/Mode Code field to 01000.

`SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:COMMAND:COUNT?` might return "XXXXX".

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat

This command specifies the data format (binary or hex) of the bit pattern for the 5-bit Word Count/Mode Code field to use in a MIL-STD-1553 bus command word search. The search condition needs to be set to `COMMAND`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:
FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:COUNT:FORMAT HEX sets the data format to HEX.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:COUNT:FORMAT? might return BINARY, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:PARity

This command specifies the parity to use in a MIL-STD-1553 bus command word search. The search condition needs to be set to COMMAND.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:PARity {0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:PARity?

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments 0

1

X sets the value to X (“don’t care”) which is the default.

ZERO sets the value to 0.

ONE sets the value to 1.

NOCARE sets the value to X (“don’t care”) which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:COMMAND:PARITY ZERO` sets the Command word parity bit to 0.

`SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:COMMAND:PARITY?` might return 0, 1 or X.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress

This command specifies the 5 bit sub-address to use in a MIL-STD-1553 bus command word sub-address search. When the sub-address value is set to 00000 or 11111 binary, it specifies that the command is a "Mode Code" command. Any other value specifies that it is a "Word Count" command. The default is all X's (don't care). The search condition needs to be set to `COMMAND`.

NOTE. *The sub-address field is used to direct data to different functions within the subsystem for values 1 - 30.*

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress
<QString>
```

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt

Arguments QString is a quoted string of up to 5 characters, where the allowable characters are 0, 1, and X.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:COMMAND:SUBADDRESS "01000"` sets the search sub-address to 01000.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:FORMat

This command specifies the data format (binary or hex) of the sub-address field bit pattern to use in a MIL-STD-1553 bus command word sub-address search. The search condition needs to be set to `COMMAND`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:
FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:
FORMat?
```

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:SUBADDRESS:FORMAT HEX` sets the data format to `HEX`.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:SUBADRESS:FORMAT?` might return `BINARY`, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:TRBit

This command specifies to search on either a “transmit” or a “receive” signal for the transmit/receive bit (bit 9), for a MIL-STD-1553 bus command word search. The search condition needs to be set to `COMMAND`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:TRBit
{RX|TX|X|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:TRBit?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt

Arguments `TX` (logic 1) directs the instrument to search for a TX or "transmit" from a remote terminal .

`RX` (logic 0) directs the instrument to search for an RX or "receive" from a remote terminal.

X, DONTCare indicates “don’t care".

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:TRBIT TX directs the instrument to search for a TX or “transmit” from a remote terminal.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:COMMAND:TRBIT? might return TX.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

This command specifies which word type or condition within a MIL-STD-1553 bus word to search on.

NOTE. *There are three types of MIL-STD-1553 (Aircraft) bus words: Command, Status, and Data, all sharing a common structure twenty bits in length. The first three bits are used as a synchronization field. The last bit is the odd parity of the previous 16 bits.*

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition {SYNC|COMMAND|STATus|DATA|TIMe|ERRor}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition?

Related Commands

Most of the other SEARCH:SEARCH1:TRIGger:A:BUS:B<x>:MIL1553B commands are impacted by the setting of this command.

TRIGger:A:BUS:MIL1553B:CONDition

BUS:B<x>:MIL1553B:SOUrce

Arguments

SYNC refers to the 3-bit sync pulse that precedes each word.

COMMAND is one of 3 16-bit word types. It specifies the function that a remote terminal is to perform.

STATus is one of 3 16-bit word types. Remote terminals respond to valid message transmissions via status words.

DATA is one of 3 16-bit word types.

TIMe specifies to search for either the RT (remote terminal response time), or the IMG (Inter-message Gap). Use the commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:QUALifier, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit, and SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit to specify the time parameters.

ERRor specifies to search for a signaling error. (You can specify which type of error – Parity, Sync, Manchester or Non-contiguous Data – by using the command SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:ERRTYPE.)

NOTE. *Use the SEARCH:SEARCH1:TRIGger:A:BUS:B<x>:MIL1553B:CONDition:STATus:BIT commands to set the following bits:*

9 — message error
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME

10 — instrumentation
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR

11 — SRQ (service request)
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ

15 — BCR (broadcast command received)
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR

16 — busy
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY

17 — subsystem flag
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF

18 — DBCA (dynamic bus control acceptance)
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA

19 — terminal flag
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:CONDITION STAT sets the word type to search for to STATUS.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:CONDITION? might return DATA.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:FORMat

This command specifies the format (binary or hex) of the bit pattern to use for a MIL-STD-1553 bus data word search. The search condition needs to be set to DATA.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:FORMat
{BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:PARity, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:DATA:FORMAT HEX sets the data format to HEX.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:DATA:FORMAT? might return BINARY, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:PARity

This command specifies the parity to use when searching on a MIL-STD-1553 bus data word parity bit. The search condition must be set to DATA.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:PARity
{0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:PARity?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:VALue

Arguments

0

1

X sets the value to X (“don’t care”) which is the default.

ZERO sets the value to 0.

ONE sets the value to 1.

NOCARE sets the value to X (“don’t care”) which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:DATA:PARITY 1 sets the data parity bit to be used in a MIL-STD-1553 bus search to 1.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:DATA:PARITY? might return 0, indicating that the data parity bit to be used in a MIL-STD-1553 bus search is set to 0.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:VALue

This command specifies the value to use for a MIL-STD-1553 bus data word search. This is a 16-bit field. The default is all X’s (don’t care). The search condition must be set to DATA.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:PARity

Arguments

QString is a quoted string of up to 16 characters, where the allowable characters are 0, 1, and X.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:DATA:VALUE "01000" sets the data binary pattern to be used in a MIL-STD-1553 bus search to 01000.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:DATA:VALUE? might return XXXXXXXXXXX01000.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:ERRTYPE

This command specifies the signaling error type to use for a MIL-STD-1553 bus data word search: Parity, Sync, Manchester, or Data. The search condition must be set to ERRor.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:ERRTYPE
{PARity|SYNC|MANCHester|DATA}
```

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:ERRTYPE?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

PARity – an incorrect parity setting.

SYNc – the high to low, or low to high transition doesn't happen in the middle of the sync time as it should.

MANCHester – no transition in a bit time.

DATA – a non-contiguous data error.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:ERRTYPE PARITY sets the error type to be used in the search to PARITY.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:ERRTYPE? might return SYNC.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when searching on the MIL-STD-1553 status address field. The search condition needs to be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:
FORMat {BINary|HEX}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:
FORMat?
```

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:STATUS:ADDRESS:FORMAT HEX` sets the data format to `HEX`.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:STATUS:ADDRESS:FORMAT?` might return `BINARY`, indicating that the data format is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier

This command specifies the qualifier to use when searching on the MIL-STD-1553 status address field. The default is EQUAL. The search condition must be set to `STATUS`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:
MIL1553B:STATUS:ADDRess:QUALifier
{LESSthan|MOREthan|EQual|UNEQual|LESSEQual|MOREEQual}
```

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:
QUALifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue

Arguments

LESSthan sets the Status Address qualifier to less than.

MOREthan sets the Status Address qualifier to greater than.

EQual sets the Status Address qualifier to equal.

UNEQual sets the Status Address qualifier to not equal.

LESSEQual sets the Status Address qualifier to less than or equal.

MOREEQual sets the Status Address qualifier to greater than or equal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:QUALIFIER MORETHAN sets the qualifier to be used with the address field to greater than.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:QUALIFIER? might return MORETHAN.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue

This command specifies the value of the 5-bit remote terminal address to use when searching on the MIL-STD-1553 status address field. The default is all X's (don't care). The search condition must be set to STATUS.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue <QString>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier

Arguments

QString is a quoted string of up to 5 characters, where the allowable characters are 0, 1, and X.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:VALUE "01000" sets the value of the 5-bit remote terminal address to be used in the search.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:VALUE? might return “01000”.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR

This command specifies the status word broadcast command received (BCR) bit value (bit 15) to use when searching on the MIL-STD-1553 status bit field. The default is all X’s (don’t care). The search condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR {0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR?

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:BCR NOCARE sets the BCR bit search value to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:BCR? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY

This command specifies the status word busy bit value (bit 16) to use when searching on the MIL-STD-1553 status bit field. The default is all X’s (don’t care). The search condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY
{0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

`0, ZERO` sets the value to 0.

`1, ONE` sets the value to 1.

`X, NOCARE, DONTCare` sets the value to X ("don't care"), which is the default.

`OFF` sets the value to 0.

`ON` sets the value to 1.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:BUSY NOCARE` sets the status word busy bit search value to X.

`SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:BUSY?` might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA

This command specifies the status word dynamic bus control acceptance (DBCA) bit value (bit 18) to use when searching on the MIL-STD-1553 status bit field. The default is all X's (don't care). The search condition must be set to `STATUS`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA
{0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:DBCA NOCARE sets the status word dynamic bus control acceptance (DBCA) bit search value to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:DBCA? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR

This command specifies the status word instrumentation bit value (bit 10) to use when searching on the MIL-STD-1553 status bit field. The default is all X’s (don’t care). The search condition must be set to STATUS.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR {0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR?

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:INSTR NOCARE sets the status word status word instrumentation bit search value to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:INSTR? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME

This command specifies the status word message error bit value (bit 9) to use when searching on the MIL-STD-1553 status bit field. The default is all X's (don't care). The search condition must be set to STATUS.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME {0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME?

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X ("don't care"), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:ME NOCARE sets the status word message error bit search value to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:ME? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity

This command specifies the parity to use when searching on the MIL-STD-1553 status bit field. The default is all X's (don't care). The search condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity
{0|1|X|ZERo|ONE|NOCARE|DONTCare|ON|OFF}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity?
```

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X ("don't care"), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:STATUS:BIT:PARITY sets the parity to be used in the trigger to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIL1553B:STATUS:BIT:PARITY? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ

This command specifies the status word service request (SRQ) bit value (bit 11) to use when searching on the MIL-STD-1553 status bit field. The default is all X's (don't care). The search condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ
{0|1|X|ZERo|ONE|NOCARE|OFF|ON}
```

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ?

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:SRQ NOCARE sets the status word SRQ bit search value to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:SRQ? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF

This command specifies the status word subsystem flag bit value (bit 17) to use when searching on the MIL-STD-1553 status bit field. The default is all X’s (don’t care). The search condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF {0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF?

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:SUBSF NOCARE sets the status word subsystem flag bit search value to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:SUBSF? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF

This command specifies the status word terminal flag bit value (bit 19) to use when searching on the MIL-STD-1553 status bit field. The default is all X's (don't care). The search condition must be set to STATUS.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF {0|1|X|ZERo|ONE|NOCARE|OFF|ON}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF?

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X ("don't care"), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:TF NOCARE sets the status word terminal flag bit search value to X.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:TF? might return 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit

This command specifies either the minimum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit, or else it specifies the minimum inter-message gap (IMG). The search condition must be set to TIMe

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit

Arguments <NR3> is a floating point number that specifies either the minimum remote terminal response time (RT) or the inter-message gap (IMG) in seconds.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:TIME:LESSLIMIT 3.0000E-6 would set either the minimum remote terminal response time (RT) or the minimum inter-message gap (IMG) to 3.00 µS.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:TIME:LESSLIMIT? might return 2.0000e-6, indicating that the minimum RT or IMG has been set to 2.00 µS.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit

This command specifies either the maximum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit, or it specifies the maximum inter-message gap (IMG). The search condition must be set to TIMe,

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit

Arguments

<NR3> is a floating point number that specifies either the maximum remote terminal response time (RT) or the inter-message gap (IMG) in seconds.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:TIME:MORELIMIT 80.0000E-6 would set either the maximum remote terminal response time (RT) or the maximum inter-message gap (IMG) to be used in the search to 80.0 µS.

SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:TIME:MORELIMIT? might return 100.0000e-6, indicating that the maximum RT or IMG has been set to 100.00 µS.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:QUALifier

This command specifies the qualifier to use when triggering on either the remote terminal response time (RT) or the inter-message gap (IMG). The search condition must be set to TIMe,

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:QUALifier {LESSthan|MOREthan|INrange|OUTrange}

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:QUALifier?

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit

Arguments

LESSthan sets the Time qualifier to less than minimum.

MOREthan sets the Time qualifier to greater than maximum

INrange sets the Time qualifier to inside range.

OUTrange sets the Time qualifier to out of range.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:TIME:QUALIFIER LESSTHAN` sets the qualifier to less than minimum.

`SEARCH:SEARCH1:TRIGGER:A:BUS:B1:MIL1553B:TIME:QUALIFIER?` might return `OUTRANGE`.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:CONDition

This command sets or queries the search condition for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:CONDition
{ESCMode|EOT|ERROR|LONG|SHORT|SOT|STOP|WARNing}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:CONDition?
```

Arguments

`ESCMode` specifies the search condition as Escape Mode.

`EOT` specifies the search condition as End Of Transmission.

`ERROR` specifies the search condition as Error.

`LONG` specifies the search condition as Long Packet.

`SHORT` specifies the search condition as Short Packet.

`SOT` specifies the search condition as Start Of Transmission.

`STOP` specifies the search condition as Stop.

`WARNing` specifies the search condition as Warning.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:CONDITION EOT` sets the search condition for the MIPI CSI2 bus trigger search 1 to EOT.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:CONDITION?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:CONDITION LONG`, indicating that the search condition for MIPI CSI2 bus trigger search 1 is set to LONG.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTLong

This command sets or queries the data type for long packets for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTLong
{ANY|BLANK|EMBEDded|NULL|RAW10|RAW12|RAW14|RAW6|RAW7|RAW8|
RGB444|RGB555|RGB565|RGB666|RGB888|USER8B1|USER8B2|USER8B3|
USER8B4|USER8B5|USER8B6|USER8B7|USER8B8|YUV420B10|YUV420C10|
YUV420B8|YUV420C8|YUV420L8|YUV422B10|YUV422B8}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTLong?
```

Arguments

`ANY` specifies the data type as Any.

`BLANK` specifies the data type as Blank.

`EMBEDded` specifies the data type as Embedded.

`NULL` specifies the data type as Null.

`RAW10` specifies the data type as RAW10.

`RAW12` specifies the data type as RAW12.

`RAW14` specifies the data type as RAW14.

`RAW6–RAW8` specifies the data type as RAW6, RAW7, or RAW8.

`RGB444` specifies the data type as RGB444.

`RGB555` specifies the data type as RGB555.

`RGB565` specifies the data type as RGB565.

`RGB666` specifies the data type as RGB666.

`RGB888` specifies the data type as RGB888.

`USER8B1–USER8B8` specifies the data type as User 8-bit 1, User 8-bit 2, User 8-bit 3, User 8-bit 4, User 8-bit 5, User 8-bit 6, User 8-bit 7, User 8-bit 8.

`YUV420B10` specifies the data type as YUV420 10-bit.

`YUV420C10` specifies the data type as YUV420 10-bit (Chroma Shifted Pixel Sampling).

`YUV420B8` specifies the data type as YUV420 8-bit.

YUV420C8 specifies the data type as YUV420 8-bit (Chroma Shifted Pixel Sampling).

YUV420L8 specifies the data type as YUV420 8-bit (legacy).

YUV422B10 specifies the data type as YUV422 10-bit.

YUV422B8 specifies the data type as YUV422 8-bit.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:DTLONG EMBEDDED sets the data type for long packets for MIPI CSI2 bus trigger search 1 to EMBEDded.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:DTLONG? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:DTLONG NULL, indicating that the data type for long packets for MIPI CSI2 bus trigger search 1 is set to NULL.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTShort

This command sets or queries the data type for short packets for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTShort
{ANY|FRAMEEnd|FRAMEStart|GSP1|GSP2|GSP3|GSP4|GSP5|
GSP6|GSP7|GSP8|LINEEnd|LINEStart}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTShort?
```

Arguments

ANY specifies the data type as Any.

FRAMEEnd specifies the data type as Frame End.

FRAMEStart specifies the data type as Frame Start.

GSP1–GSP8 specifies the data type as GSP1 to GSP8, where GSP refers to General Short Packet.

LINEEnd specifies the data type as Line End.

LINEStart specifies the data type as Line Start.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:DTSHORT FRAMEEND` sets the data type for short packets for MIPI CSI2 bus trigger search 1 to FRAMEEnd.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:DTSHORT?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:DTSHORT ANY`, indicating that the data type for short packets for MIPI CSI2 bus trigger search 1 is set to ANY.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ERRor

This command sets or queries the error type for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ERRor
{ANY|CHECKsumerror|ECCError}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ERRor?
```

Arguments `ANY` specifies the error type as Any.

`CHECKsumerror` specifies the error type as CHECKsumerror.

`ECCError` specifies the error type as ECCError.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:ERROR ANY` sets the error type for MIPI CSI2 bus trigger search 1 to ANY.

`SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:ERROR?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:ERROR ECCERROR`, indicating that the error type for MIPI CSI2 bus trigger search 2 is set to ECCError.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ESCape

This command sets or queries the Escape command for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ESCape
{ANY|LPDATA|RESET|ULTRALP}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ESCape?
```

Arguments

ANY specifies the Escape command as Any.

LPDATA specifies the Escape command as Low Power Data Transmission.

RESET specifies the Escape command as Reset.

ULTRALP specifies the Escape command as Ultra Low Power State.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:ESCAPE ANY sets the Escape command for MIPI CSI2 bus trigger search 1 to ANY.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:ESCAPE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:ESCAPE LPDATA, indicating that the Escape command for MIPI CSI2 bus trigger search 2 is set to LPDATA.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:OFFset:VALue

This command sets or queries the offset value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:OFFset:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:OFFset:VALue?
```

Arguments

<NR3> specifies an offset value.

DONTCare specifies the offset value as Don't Care.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:OFFSET:VALUE 599 sets the offset value for the MIPI CSI2 bus trigger search to 599.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:OFFSET:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:OFFSET:VALUE DONTCARE, indicating that the offset value for MIPI CSI2 bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:FORMat

This command sets or queries the format of the packet data for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:FORMat?
```

Arguments

`BINary` specifies the data packet format as Binary.

`HEXadecimal` specifies the data packet format as Hexadecimal.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PACKETDATA:FORMAT BIN` sets the packet data format for MIPI CSI2 bus trigger search 1 to binary.

`SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PACKETDATA:FORMAT?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PACKETDATA:FORMAT HEX`, indicating that the packet data format for MIPI CSI2 bus trigger search 2 is set to hexadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:VALue

This command sets or queries the value of the packet data string for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:VALue
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:VALue?
```

Arguments

<string> specifies the 16-bit packet data string. The valid characters are 0, 1, and X when the format is set to Binary; and 0-9, A-F, and X when the format is set to hexadecimal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PACKETDATA:VALUE "100" sets the value of the packet data string to "XXXXXXXXXXXXX100" for MIPI CSI2 bus trigger search 1 when the packet data format is set to binary.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PACKETDATA:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PACKETDATA:VALUE "X100", indicating that the value of the packet data string for MIPI CSI2 bus trigger search 2 is set to "100," in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat

This command sets or queries the generic payload format for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:VALue

Arguments

BINary specifies the generic payload format as Binary.

HEXadecimal specifies the generic payload format as Hexadecimal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:FORMAT BINARY sets the generic payload format for MIPI CSI2 bus trigger search 1 to BINary.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:FORMAT? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:FORMAT HEX, indicating that the generic payload format for MIPI CSI2 bus trigger search 2 is set to HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe

This command sets or queries the generic payload size of the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:VALue

Arguments <NR3> specifies a number from 1 to 16 that sets the generic payload size in bytes.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:SIZE 12 sets the generic payload size of the MIPI CSI2 bus trigger search 1 to 12 bytes.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:SIZE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:SIZE 16, indicating that the generic payload size of MIPI CSI2 bus trigger search 1 is set to 16 bytes.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:VALue

This command sets or queries the generic payload value string of the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:VALue
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe

Arguments <string> specifies the generic payload value. The valid characters are 0, 1, and X for values in binary format; and A-F, 0-9, and X for values in hexadecimal format.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:VALUE "100" sets the generic payload value to "XXXXXXXXXXXXX100" for MIPI CSI2 bus trigger search 1 in binary format.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PAYLOAD:VALUE "XX101", indicating that the generic payload value for MIPI CSI2 bus trigger search 2 is set to "XX101" in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PIXel:VALue

This command sets or queries the pixel value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PIXel:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PIXel:VALue?
```

Arguments <NR3> specifies a number that represents the pixel value.

DONTCare specifies the pixel value as Don't Care.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:PIXEL:VALUE 350 sets the pixel value for MIPI CSI2 bus trigger search 1 to 350.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PIXEL:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:PIXEL:VALUE DONTCARE, indicating that the pixel value for MIPI CSI2 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:QUAlifier

This command sets or queries the RAW qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:QUAlifier?
```

Arguments

`EQUal` specifies the RAW qualifier as Equal.

`LESSThan` specifies the RAW qualifier as Less Than.

`MOREThan` specifies the RAW qualifier as More Than.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RAW:QUALIFIER EQUAL` sets the RAW qualifier for MIPI CSI2 bus trigger search 1 to EQUal.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RAW:QUALIFIER?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RAW:QUALIFIER LESSTHAN`, indicating that the RAW qualifier for MIPI CSI2 bus trigger search 1 is set to LESSThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:VALue

This command sets or queries the RAW value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:VALue?
```

Arguments

<NR3> specifies a number that represents the RAW value.

DONTCare specifies the RAW value as Don't Care.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RAW:VALUE 100 sets the RAW value for MIPI CSI2 bus trigger search 1 to 100.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RAW:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RAW:VALUE DONTCARE, indicating that the RAW value for MIPI CSI2 bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:QUAlifier

This command sets or queries the RGB Blue qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:
QUAlifier?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:VALue

Arguments

EQUal specifies the qualifier as Equal.

LESSThan specifies the qualifier as Less Than.

MOREThan specifies the qualifier as More Than.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RGB:BLUE:QUALIFIER EQUAL sets the RGB Blue qualifier for MIPI CSI2 bus trigger search 1 to EQUal.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:BLUE:QUALIFIER? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:BLUE:QUALIFIER LESSTHAN, indicating that the RGB Blue qualifier for MIPI CSI2 bus trigger search 2 is set to LESSThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:VALue

This command sets or queries the RGB Blue value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x>is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:QUAlifier

Arguments <NR3> specifies a number for the RGB Blue value; the range of the number is dependent on the data type.

DONTCare specifies the RGB Blue value as Don't Care.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RGB:BLUE:VALUE 125 sets the RGB Blue value to 125.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:BLUE:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPIDSIONE:RGB:BLUE:VALUE DONTCARE, indicating that the RGB Blue value for MIPI CSI2 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:QUAlifier

This command sets or queries the RGB Green qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:
QUAlifier {EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:
QUAlifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:VALue

Arguments `EQUal` specifies the qualifier as Equal.

`LESSThan` specifies the qualifier as Less Than.

`MOREThan` specifies the qualifier as More Than.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RGB:GREEN:QUALIFIER EQUAL` sets the RGB Green qualifier for MIPI CSI2 bus trigger search 1 to EQUal.

`SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:GREEN:QUALIFIER?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:GREEN:QUALIFIER LESSTHAN`, indicating that the RGB Green qualifier for MIPI CSI2 bus trigger search 2 is set to LESSThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:VALue

This command sets or queries the RGB Green value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:QUAlifier

Arguments `<NR3>` specifies a number for the RGB Green value; the range of the number is dependent on the data type.

`DONTCare` specifies the RGB Green value as Don't Care.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RGB:GREEN:VALUE 164` sets the RGB Green value for MIPI CSI2 bus trigger search 1 to 164.

`SEARCH:SEARCH21:TRIGGER:A:BUS:MIPICSITWO:RGB:GREEN:VALUE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:GREEN:VALUE`

DONTCARE, indicating that the RGB Green value for MIPI CSI2 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:QUAlifier

This command sets or queries the RGB Red qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:QUAlifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:VALue

Arguments

EQUal specifies the qualifier as Equal.

LESSThan specifies the qualifier as Less Than.

MOREThan specifies the qualifier as More Than.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RGB:RED:QUALIFIER EQUAL sets the RGB Red qualifier for MIPI CSI2 bus trigger search 1 to EQUal.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:RED:QUALIFIER? might return
:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:RED:QUALIFIER LESSTHAN, indicating that the RGB Red qualifier for MIPI CSI2 bus trigger search 2 is set to LESSThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:VALue

This command sets or queries the RGB Red value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:VALue?
```

Arguments

<NR3> specifies a number for the RGB Red value; the range of the number is dependent on the data type.

DONTCare specifies the RGB Red value as Don't Care.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:RGB:RED:VALUE 108 sets the RGB Red value for MIPI CSI2 bus trigger search 1 to 108.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:RED:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:RGB:RED:VALUE DONTCARE, indicating that the RGB Red value for MIPI CSI2 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:VC

This command sets or queries the Virtual Channel for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:VC
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:VC?
```

Arguments

<NR3> specifies a number from 0 to 3 that represents the virtual channel.

DONTCare specifies the virtual channel as Don't Care.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:VC 3 sets the virtual channel for MIPI CSI2 bus trigger search 1 to 3.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:VC? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:VC DONTCARE, indicating that the virtual channel for MIPI CSI2 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WARNing

This command sets or queries the warning type for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WARNing
{ANY|ECCWarn}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WARNing?
```

Arguments `ANY` specifies the warning type as Any.

`ECCWarn` specifies the warning type as ECC Warning.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:WARNING ANY` sets the warning type for MIPI CSI2 bus trigger search 1 to ANY.

`SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:WARNING?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:WARNING ECCWARN`, indicating that the warning type for MIPI CSI2 bus trigger search 2 is set to ECCWarn.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WC:VALue

This command sets or queries the word count value for a specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WC:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WC:VALue?
```

Arguments `<NR3>` specifies a number for the word count value.

`DONTCare` specifies the word count value as Don’t Care.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:WC:VALUE 50` sets the word count value for MIPI CSI2 bus trigger search 1 to 50.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:WC:VALUE?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:WC:VALUE DONTCARE`, indicating that the word count value for MIPI CSI2 bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:QUAlifier

This command sets or queries the YUV U qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:QUAlifier?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:VALue

Arguments

`EQUal` specifies the qualifier as Equal.

`LESSThan` specifies the qualifier as Less Than.

`MOREThan` specifies the qualifier as More Than.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:U:QUALIFIER EQUAL` sets the YUV U qualifier for MIPI CSI2 bus trigger search 1 to EQUal.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:U:QUALIFIER?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:U:QUALIFIER LESSThan`, indicating that the YUV U qualifier for MIPI CSI2 bus trigger search 1 is set to LESSThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:VALue

This command sets or queries the YUV U value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:QUAlifier

Arguments <NR3> specifies a number for the YUV U value; the range of the number is dependent on the data type.

DONTCare specifies the YUV U value as Don't Care.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:U:VALUE 104 sets the YUV U value for MIPI CSI2 bus trigger search 1 to 104.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:YUV:U:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:YUV:U:VALUE DONTCARE, indicating that the YUV U value for MIPI CSI2 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:QUAlifier

This command sets or queries the YUV V qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:QUAlifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:VALue

Arguments EQUal specifies the qualifier as Equal.

LESSThan specifies the qualifier as Less Than.

MOREThan specifies the qualifier as More Than.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:V:QUALIFIER EQUAL sets the YUV V qualifier for MIPI CSI2 bus trigger search 1 to EQUal.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:V:QUALIFIER? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:V:QUALIFIER MORETHAN, indicating that the YUV V qualifier for MIPI CSI2 bus trigger search 1 is set to MOREThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:VALue

This command sets or queries the YUV V value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:QUAlifier

Arguments

<NR3> specifies a number for the YUV V value; the range of the number is dependent on the data type.

DONTCare specifies the YUV V value as Don't Care.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:V:VALUE 1003 sets the YUV V value for MIPI CSI2 bus trigger search 1 to 1003.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:V:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:V:VALUE DONTCARE, indicating that the YUV V value for MIPI CSI2 bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:QUAlifier

This command sets or queries the YUV Y qualifier for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:QUAlifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:VALue

Arguments

`EQUal` specifies the qualifier as Equal.

`LESSThan` specifies the qualifier as Less Than.

`MOREThan` specifies the qualifier as More Than.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:Y:QUALIFIER LESSTHAN` sets the YUV Y qualifier for MIPI CSI2 bus trigger search 1 to LESSThan.

`SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:YUV:Y:QUALIFIER?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:YUV:Y:QUALIFIER EQUAL`, indicating that the YUV Y qualifier for MIPI CSI2 bus trigger search 2 is set to EQUal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:VALue

This command sets or queries the YUV Y value for the specified MIPI CSI2 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:VALue
{<NR3>|DONTCare}
```

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:VALue?

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:QUALifier

Arguments <NR3> specifies a number for the YUV Y value; the range of the number is dependent on the data type.

DONTCare specifies the YUV Y value as Don't Care.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPICSITWO:YUV:Y:VALUE 1004 sets the YUV Y value for MIPI CSI2 bus trigger search 1 to 1004.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:YUV:Y:VALUE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPICSITWO:YUV:Y:VALUE DONTCARE, indicating that the YUV Y value for MIPI CSI2 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ACKERRreport

This command sets or queries the Acknowledge and Error Report for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ACKERRreport
{ANY|CHKSUMError|CONTention|DSINR|DSIViolation|DSIVC|
ECCMBError|ECCSBError|EOTSync|ESCMODEError|FCE|HSRTError|
INVALID|LPTSError|SOTError|SOTSync}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ACKERRreport?
```

Arguments ANY matches any error condition.

CHKSUMError Checksum Error.

CONTention is the contention detected.

DSINR DSI Data Type Not Recognized.

DSIViolation DSI Protocol Violation.

DSIVC DSI Virtual Channel ID Invalid.

ECCMBError ECC Error, Multi-bit.

ECCSBError ECC Error, Single-bit.

EOTSync End of Transmission Sync.

ESCMODEError Escape Mode Entry Command Error.

FCE False Control Error.

HSRTError High-Speed Receive Timeout Error.

INVALID Invalid Transmission Length.

LPTSError Low-Power Transmit Sync Error.

SOTError Start of Transmission Error.

SOTSync Start of Transmission Sync Error.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:ACKERRREPORT SOTERROR sets the search condition for MIPI DSI1 bus trigger search 1 to SOTError.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPIDSIONE:ACKERRREPORT? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIPIDSIONE:ACKERRREPORT ANY, indicating that the search condition for MIPI DSI1 bus trigger search 1 is set to ANY.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:CONDition

This command sets or queries the search condition for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:CONDition
{ESCMode|BTA|EOT|ERROR|LONG|SHORT|SOT|STOP|WA}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:CONDition?
```

Arguments

ESCMode specifies the search condition as Escape Mode.

BTA specifies the search condition as Bus Turnaround.

EOT specifies the search condition as End of Transmission.

ERROR specifies the search condition as Error.

LONG specifies the search condition as Long Packet.

SHORT specifies the search condition as Short Packet.

SOT specifies the search condition as Start of Transmission.

STOP specifies the search condition as Low Power Stop State.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:CONDITION ERROR sets the search condition for MIPI DSI1 bus trigger search 1 to ERROR.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:CONDITION? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:CONDITION STOP, indicating that the search condition for MIPI DSI1 bus trigger search 1 is set to STOP.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DIRection

This command sets or queries the direction for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DIRection
{FORWard|REVErse|ANY}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DIRection?
```

Arguments

FORWard specifies the direction as Forward.

REVErse specifies the direction as Reverse.

ANY specifies the direction as Any.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:DIRECTION FORWARD sets the direction for MIPI DSI1 bus trigger search 1 to FORWard.

SEARCH:SEARCH2:TRIGGER:A:BUS:MIPIDSIONE:DIDRECTION? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:MIMPIDSIONE:DIRECTION ANY, indicating that the direction for MIPI DSI1 bus trigger search 2 is set to ANY.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTLong

This command sets or queries the data type for long packets for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTLong
{ANY|BLANK|DCSLONGRead|DCSLONGWrite|GLONGRead|GLONGWrite|
LPS666|NULL|PPS565|PPS666|PPS888|PPS101010|PPS121212|
YCBCR12|YCBCR16|YCBCR20|YCBCR24}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTLong?
```

Arguments

`ANY` specifies the data type as Any.

`BLANK` specifies the data type as Blank.

`DCSLONGRead` specifies the data type as DCS Long Read.

`DCSLONGWrite` specifies the data type as DCS Long Write.

`GLONGRead` specifies the data type as General Long Read Response.

`GLONGWrite` specifies the data type as General Long Write.

`LPS666` specifies the data type as Loosely Packed 18-bit RGB, 6-6-6.

`NULL` specifies the data type as Null.

`PPS565` specifies the data type as 16-bit RGB, 5-6-5.

`PPS666` specifies the data type as 18-bit RGB, 6-6-6.

`PPS888` specifies the data type as 24-bit RGB, 8-8-8.

`PPS101010` specifies the data type as 30-bit RGB, 10-10-10.

`PPS121212` specifies the data type as 36-bit RGB, 12-12-12.

`YCBCR12` specifies the data type as 12-bit YCbCr, 4:2:0.

`YCBCR16` specifies the data type as 16-bit YCbCr, 4:2:2.

`YCBCR20` specifies the data type as 20-bit YCbCr, 4:2:2.

`YCBCR24` specifies the data type as 24-bit YCbCr, 4:2:2.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:DTLONG GLONGWRITE` sets the data type for long packets for MIPI DSI1 bus trigger search 1 to GLONGWrite.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:DTLONG? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:DTLONG NULL, indicating that the data type for MIPI DSI1 bus trigger search 1 long packets is set to NULL.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTShort

This command sets or queries the data type for short packets for the MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTShort
{ACKErrorreport|ANY|COLOROFF|COLORON|DCSR|DCSSRR1|
DCSRR2|DCSSW0|DCSSW1|EOT|GR0|GR1|GR2|GSRR1|GSRR2|
GSW0|GSW1|GSW2|HSYNCEnd|HSYNCStart|MAXRETsize|SHUTDown|
TURNON|VSYNCEnd|VSYNCStart}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTShort?
```

Arguments

ACKErrorreport Acknowledge and Error report.

ANY matches any data type.

COLOROFFColor Mode Off.

COLORON Color Mode On.

DCSR DCS Read.

DCSSRR1 DCS Short Read Response, 1 byte.

DCSRR2 DCS Short Read Response, 2 bytes.

DCSSW0 DCS Short Write, no parameters.

DCSSW1 DCS Short Write, 1 parameter.

EOT End of Transmission.

GR0 Generic Read, no parameters.

GR1 Generic Read, one parameter.

GR2 Generic Read, two parameters.

GSRR1 Generic Short Read Response, 1 byte.

GSRR2 Generic Short Read Response, 2 bytes.

GSW0 Generic Short Write, no parameters.

GSW1 Generic Short Write, one parameters

GSW2 Generic Short Write, two parameters.

HSYNCEnd HSync End.

HSYNCStart HSync Start.

MAXRETsize Set Maximum Return Packet Size.

SHUTDown Shut Down Peripheral.

TURNON Turn On Peripheral.

VSYNCEnd VSync End

VSYNCStart VSync Start.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:MIPIDSIOne:DTShort SHUTDOWN sets the data type for MIPI DSI1 bus trigger search 1 short packets to SHUTDown.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:DTSHORT? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:DTSHORT ANY, indicating that the data type for MIPI DSI1 bus trigger search 1 short packets is set to ANY.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ERRor

This command sets or queries the error type for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ERRor
{ANY|CHECKsumerror|ECCError}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ERRor?
```

Arguments ANY specifies the error type as Any.

CHECKsumerror specifies the error type as Check Sum Error.

ECCError specifies the error type as ECC Error.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:ERROR CHECKsumerror sets the error type for MIPI DSI1 bus trigger search 1 to CHECKsumerror.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:ERROR? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:ERROR ECCError, indicating that the error type for MIPI DSI1 bus trigger search 1 is set to ECCError.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ESCape

This command sets or queries the Escape command for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ESCape
{ACK|LPDATA|RESET|ANY|TEAR|ULTRALP}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ESCape?
```

Arguments

ACK specifies the Escape command as Ack (positive acknowledgement).

LPDATA specifies the Escape command as Low Power Data Transmission.

RESET specifies the Escape command as Reset.

ANY specifies the Escape command as Any.

TEAR specifies the Escape command as Tearing Effect.

ULTRALP specifies the Escape command as Ultra Low Power State.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:ESCAPE RESET sets the Escape command for MIPI DSI1 bus trigger search 1 to RESET.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:ESCAPE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:ESCAPE ACK, indicating that the Escape command for MIPI DSI1 bus trigger search 1 is set to ACK.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue

This command sets or queries the offset value for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue?
```

Arguments <NR3> specifies an offset value.

DONTCare specifies the offset value as Don't Care.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue 34445 sets the offset value for MIPI DSI1 bus trigger search 1 to 34445.

SEARCH:SEARCH1:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue? might return :SEARCH:SEARCH1:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue DONTCare, indicating the offset value for MIPI DSI1 bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:FORMat

This command sets or queries the format of the packet data for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUEOne, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUETwo

Arguments BINary specifies the format of the packet data as Binary.

HEXadecimal specifies the format of the packet data as Hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:FORMAT BINARY sets the packet data format for MIPI DSI1 bus trigger search 1 to BINary.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:FORMAT? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:FORMAT HEX, indicating that the format for MIPI DSI1 bus trigger search 1 packet data is set to HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUEOne

This command sets or queries the value of the first byte (8 bits) of a packet data string for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:
VALUEOne <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:
VALUEOne?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:FORMat, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUETwo

Arguments <string> specifies the first byte of the packet data string. The valid characters are 0, 1, and X when the format is set to binary; and 0-9, A-F, and X when the format is set to hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:VALUEONE "09" sets the first byte of the packet data string for MIPI DSI1 bus trigger search 1 to "09" when the packet data format is hexadecimal.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:VALUEONE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:VALUEONE "00001001", indicating that the value of the first byte of the packet data string for MIPI DSI1 bus trigger search 1 is "00001001" when the packet data format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUETwo

This command sets or queries the value of the second byte (8 bits) of a packet data string for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:
VALUETwo <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:
VALUETwo?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:FORMat, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUEOne

Arguments <string> specifies the second byte of a packet data string. The valid characters are 0, 1, and X when the format is set to binary; and 0-9, A-F, and X when the format is set to hexadecimal.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:VALUETWO "01" sets the second byte of a packet data string for MIPI DSI1 bus trigger search 1 to "01" when the packet data format is hexadecimal.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:VALUETWO? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PACKETDATA:VALUETWO "XXXXXX01", indicating that the second byte of a packet data string for MIPI DSI1 bus trigger search 1 is "XXXXXX01" when the packet data format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:FORMat

This command sets or queries the generic payload format for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:SIZe,
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:VALue

Arguments

`BINary` specifies the generic payload format as Binary.

`HEXadecimal` specifies the generic payload format as Hexadecimal.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:FORMAT BINARY` sets the generic payload format for MIPI DSI1 bus trigger search 1 to BINary.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:FORMAT?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:FORMAT HEX`, indicating that the generic payload format for MIPI DSI1 bus trigger search is set to HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:SIZe

This command sets or queries the generic payload size of the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:SIZe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:SIZe?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:FORMat,
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:VALue

Arguments

<NR3> specifies a number from 1 to 16, that sets the size of the generic payload in bytes.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:SIZE 5` sets the size of MIPI DSI1 bus trigger search 1 payload to 5 bytes.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:SIZE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:SIZE 16, indicating that the payload size of MIPI DSI1 bus trigger search 1 is set to 16 bytes.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:VALue

This command sets or queries the generic payload value string of the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:VALue
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat, SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe

Arguments <string> specifies the payload value. The valid characters are 0, 1, and X for values in binary format; and A-F, 0-9, and X for values in hexadecimal format.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:VALUE "A1234" sets the payload value string for MIPI DSI1 bus trigger search 1 to "A1234" when the payload format is hexadecimal.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PAYLOAD:VALUE "XXXXXXXXXXXXXXXXXX11", indicating that the payload value string for MIPI DSI1 bus trigger search 1 is set to "XXXXXXXXXXXXXXXXXX11" in binary format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PIXel:VALue

This command sets or queries the pixel value for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PIXel:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PIXel:VALue?
```

Arguments <NR3> specifies a number that represents the pixel value.

DONTCare specifies the pixel value as Don't Care.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PIXEL:VALUE 55 sets the pixel value for MIPI DSI1 bus trigger search 1 to 55.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PIXEL:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:PIXEL:VALUE 65535, indicating that the pixel value for MIPI DSI1 bus trigger search 1 is set to 65535.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:QUAlifier

This command sets or queries the RGB Blue qualifier for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:
QUAlifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:VALue

Arguments EQUal specifies the qualifier as Equal.

LESSThan specifies the qualifier as Less Than.

MOREThan specifies the qualifier as More Than.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:BLUE:QUALIFIER EQUAL` sets the RGB Blue qualifier for the specified MIPI DSI1 bus trigger search to EQUal.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:BLUE:QUALIFIER?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:BLUE:QUALIFIER LESSTHAN`, indicating that the RGB Blue qualifier for the specified MIPI DSI1 bus trigger search is set to LESSThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:VALue

This command sets or queries the RGB Blue value for the for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:QUAlifier

Arguments

`<NR3>` specifies a number for the RGB Blue value; the range is dependent on the data type.

`DONTCare` specifies the RGB Blue value as Don't Care.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:BLUE:VALUE 125` sets the RGB Blue value for MIPI DSI1 bus trigger search 1 to 125.

`SEARCH:SEARCH2:TRIGGER:A:BUS:MIPIDSIONE:RGB:BLUE:VALUE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPIDSIONE:RGB:BLUE:VALUE DONTCARE`, indicating that the RGB Blue value for MIPI DSI1 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:QUAlifier

This command sets or queries the RGB Green qualifier for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:
QUAlifier {EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:
QUAlifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:VALue

Arguments

`EQUal` specifies the qualifier as Equal.

`LESSThan` specifies the qualifier as Less Than.

`MOREThan` specifies the qualifier as More Than.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:GREEN:QUALIFIER EQUAL` sets the RGB Green qualifier for MIPI DSI1 bus trigger search 1 to EQUal.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:GREEN:QUALIFIER?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:GREEN:QUALIFIER LESSTHAN`, indicating that the RGB Green qualifier for MIPI DSI1 bus trigger search 1 is set to LESSThan.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:VALue

This command sets or queries the RGB Green value for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:QUAlifier

Arguments

<NR3> specifies a number for the RGB Green value; the range is dependent on the data type.

DONTCare specifies the RGB Green value as Don't Care.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:GREEN:VALUE 255 sets the RGB Green value for MIPI DSI1 bus trigger search 1 to 255.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:GREEN:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:GREEN:VALUE DONTCARE, indicating that the RGB Green value for MIPI DSI1 bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:QUAlifier

This command sets or queries the RGB Red qualifier for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:QUAlifier?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:VALue

Arguments

EQUal specifies the qualifier as Equal.

LESSThan specifies the qualifier as Less Than.

MOREThan specifies the qualifier as More Than.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:RED:QUALIFIER MORETHAN` sets the RGB Red value for MIPI DSI1 bus trigger search 1 to MOREThan.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:RED:QUALIFIER?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:RED:QUALIFIER EQUAL`, indicating that the RGB Red qualifier for MIPI DSI1 bus trigger search 1 is set to EQUal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:VALue

This command sets or queries the RGB Red value for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:VALue {<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:QUAlifier

Arguments

`<NR3>` specifies a number for the RGB Red value; the range is dependent on the data type.

`DONTCare` specifies the RGB Red value as Don't Care.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:VALUE 25` sets the RGB Red value for MIPI DSI1 bus trigger search 1 to 25.

`SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:RGB:VALUE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:MIPIDSIONE:RGB:VALUE DONTCARE`, indicating that the RGB Red value for MIPI DSI1 bus trigger search 2 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:VC

This command sets or queries the Virtual Channel for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:VC
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:VC?
```

Arguments <NR3> specifies a Virtual Channel from 0 to 3.

DONTCare specifies the Virtual Channel as Don't Care.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:VC 3 sets the virtual channel for MIPI DSI1 bus trigger search 1 to 3.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:VC? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:VC 1, indicating that the virtual channel for MIPI DSI1 bus trigger search 1 is set to 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WARNing

This command sets or queries the warning type for a specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WARNing
{ANY|ECCWarn}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WARNing?
```

Arguments ANY specifies the warning type as Any.

ECCWarn specifies the warning type as ECCWarn.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:WARNING ECCWARN sets the warning type for MIPI DSI1 bus trigger search 1 to ECCWarn.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:WARNING? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:WARNING ANY, indicating that the warning type for MIPI DSI1 bus trigger search 1 is set to ANY.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WC:VALue

This command sets or queries the word count value for the specified MIPI DSI1 bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WC:VALue
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WC:VALue?
```

Arguments <NR3> specifies a number for the word count value.

DONTCare specifies the word count value as Don't Care.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:WC:VALUE 45 sets the word count value for MIPI DSI1 bus trigger search 1 to 45.

SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:WC:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:WC:VALUE DONTCARE, indicating that the word count value for MIPI DSI1 bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:QUAlifier

This command sets or queries MIPI DSI-1 CB qualifier for the specified search. The search is specified by x. The value of x can range from 1 to 8.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:
QUAlifier?
```

Arguments EQUal sets the qualifier to equal.

LESSThan sets the qualifier to less than.

MOREThan sets the qualifier to more than.

Examples

`SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:QUAlifier lessthan` sets the qualifier to lessthan.

`SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:QUAlifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:YCBCR:CB:QUALIFIER EQUAL`, indicating that the qualifier is set to equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue

This command sets or queries MIPI DSI-1 CB value for the specified search. The search is specified by x. The value of x can range from 1 to 8.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue {<nr3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue?
```

Arguments

`<nr3>` specifies the CB value.

`DONTCare` specifies the CB values is don't care.

Examples

`SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue 2` sets the CB value to 2.

`SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:YCBCR:CB:VALUE DONTCARE`, indicating the CB value is don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:QUAlifier

This command sets or queries MIPI DSI-1 CR qualifier for the specified search. The search is specified by x. The value of x can range from 1 to 8.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:
QUAlifier?
```

Arguments EQUal sets the qualifier to equal.

LESSThan sets the qualifier to less than.

MOREThan sets the qualifier to more than.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:QUAlifier lessthan` sets the qualifier to less than.

`SEARCH:SEARCH1:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:QUAlifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:YCBCR:CR:QUALIFIER EQUAL`, indicating the CR qualifier is set to equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue

This command sets or queries MIPI DSI-1 CR value for the specified search. The search is specified by x. The value of x can range from 1 to 8.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue
{<nr3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue?
```

Arguments `<nr3>` specifies the CR value.

`DONTCare` specifies the CR values is don't care.

Examples `SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue 2` sets the CR value to 2.

`SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue?` might return

:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:YCBCR:CR:VALUE DONTCARE, indicating the CR value is don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier

This command sets or queries MIPI DSI-1 Y qualifier for the specified search. The search is specified by x. The value of x can range from 1 to 8.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier
{EQUal|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier?
```

Arguments EQUal sets the qualifier to equal.

LESSThan sets the qualifier to less than.

MOREThan sets the qualifier to more than.

Examples `SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier LessThan` sets the qualifier to less than.

`SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:YCBCR:Y:QUALIFIER EQUAL`, indicating the qualifier is set to equal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue

This command sets or queries MIPI DSI-1 Y value for the specified search. The search is specified by x. The value of x can range from 1 to 8.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue
{<nr3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue?
```

Arguments

<nr3> specifies the Y value.

DONTCare specifies the Y values is don't care.

Examples

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue 2 sets the Y value to 2.

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:MIPIDSIONE:YCBCR:Y:VALUE DONTCARE, indicating the Y value is set to don't care.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:FORMat

This command sets or queries the data format for a parallel bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:FORMat {BINary|HEXadecimal|SYMBolic}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:FORMat?

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:VALue

Arguments

BINary specifies the data format as Binary.

HEXadecimal specifies the data format as Hexadecimal.

SYMBolic specifies the data format as Symbolic.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:PARALLEL:DATA:FORMAT SYMBOLIC sets the data format for parallel bus trigger search 1 to SYMBolic.

SEARCH:SEARCH1:TRIGGER:A:BUS:PARALLEL:DATA:FORMAT? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PARALLEL:DATA:FORMAT HEX, indicating that the data format for parallel bus trigger search 1 is HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:VALue

This command sets or queries the data value string for a parallel bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:FORMat

Arguments <string> specifies the data value in a valid format. Valid characters are 0-9,

Examples SEARCH:SEARCH2:TRIGGER:A:BUS:PARALLEL:DATA:VALUE "XXXXXXXX" sets the string data value to "XXXXXXXX" in binary format.

SEARCH:SEARCH1:TRIGGER:A:BUS:PARALLEL:DATA:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PARALLEL:DATA:VALUE "XXXXXXXXXXXXXXX10010", indicating that the data value string is set to "XXXXXXXXXXXXXXX10010" in binary format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR

This command sets or queries the PCIE bus search trigger character. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR <STRING>
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>,

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

Arguments <STRING> specifies the trigger character. The string can be a mnemonic, K, or D code character.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CHARacter:CHAR "COM" sets the trigger character to COM.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CHARacter:CHAR? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:CHARACTER:CHAR "K28.5", indicating the trigger character is set to K28.5 with nothing or neutral disparity.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

This command sets or queries the PCIE bus search character symbol minus. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

Arguments <string> specifies the character symbol minus.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus "10000001" sets the character symbol minus to 10000001.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus? might return

:SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:CHARACTER:SYMBOL:MINUS "0011111010", indicating the character symbol minus is 0011111010.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

This command sets or queries the PCIE bus search trigger character symbol plus. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

Arguments <string> specifies the character symbol plus.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS “10110010” sets the character symbol plus to 10110010.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS? might return
:SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:CHARACTER:SYMBOL:PLUS "1100000101", indicating the character symbol plus is set to 1100000101.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CONDition

This command sets or queries the PCIE bus search trigger condition. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CONDition
{ANYControl|CHARacter|ERROR|PATtern|LONGORDERedset|SYNCHeader}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CONDition?
```

Arguments Arguments are the available search trigger conditions.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CONDition ERROR` sets the search trigger condition to error.

`SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:CONDition?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:CONDITION PATTERN`, indicating the search trigger condition is pattern.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:DISParity

This command sets or queries the PCIE bus search trigger disparity. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:DISParity
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:DISParity?
```

Arguments `NEGAtive` specifies negative disparity.

`POSITIVe` specifies positive disparity.

`EITher` specifies either disparity.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:DISParity NEGATIVE` sets the trigger disparity to negative.

`SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:DISParity?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:DISPARITY EITHER`, indicates either positive or negative disparity.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:CHAR

This command sets or queries the PCIE bus search trigger error character. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:CHAR
{<NR1>|OFF|ON}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:CHAR?
```

Arguments <NR1> = 0 disables the function; any other value enables it.

OFF disables the function.

ON enables the function.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:CHAR 0 disables the function.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:CHAR? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:ERROR:CHAR 1, indicating the function is on.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:DISParity

This command sets or queries the PCIE search trigger error disparity. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:DISParity
{<NR1>|OFF|ON}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:DISParity?
```

Arguments <NR1> = 0 disables the function; any other value enables it.

OFF disables the function.

ON enables the function.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:DISParity 0 disables the function.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:DISParity? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:ERROR:DISPARITY 1, indicating the function is on.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:ORDERedset

This command sets or queries the PCIE bus search trigger error ordered set. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:ORDERedset
{<NR1>|OFF|ON}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:ORDERedset?
```

Arguments <NR1> = 0 disables the function; any other value enables it.

OFF disables the function.

ON enables the function.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:ORDERedset 0 disables the function.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:ORDERedset? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:ERROR:ORDEREDSET 1, indicating the function is enabled.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:SYNC

This command sets or queries the triggering on SYNC error state.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:SYNC
{<NR1>|OFF|ON}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:SYNC?
```

Arguments

<NR1> = 1 enables the triggering on a SYNC error. Any other character disables the triggering.

OFF disables triggering on a SYNC error.

ON enables triggering on a SYNC error.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:SYNC ON enables triggering on a SYNC error in search 1.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:SYNC? might return SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:ERROR:SYNC 1 indicating that triggering on a SYNC error is enabled for search 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:FORMat

This command sets or queries the PCIE search trigger format. The search number is specified by x.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:FORMat
{ORDERedset|SYMbol|CHAR}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:FORMat?
```

Arguments

Arguments are the available formats.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:FORMat SYMBOL sets the format to symbol.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:FORMat? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:FORMAT ORDEREDSET, indicating the format is orderedset.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:LONGORDERedset

This command sets or queries the PCIE bus search trigger long ordered set. The search number is specified by x.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:LONGORDERedset
{CTRLSKP|EI|EIE|FTS|SDS|SKP|TS1|TS2}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:LONGORDERedset?
```

Arguments

`CTRLSKP` is the Control SKIP ordered set (Gen4).

`EI` is the Electrical Idle ordered set (Gen3).

`EIE` is the Electrical Idle Exit ordered set.

`FTS` is the Fast Training Sequence ordered set (Gen3).

`SDS` is the Start of Data Stream ordered set (Gen3).

`SKP` is the SKIP ordered set (Gen3).

`TS1` is the Training Sequence 1 ordered set.

`TS2` is the Training Sequence 2 ordered set.

Examples

`SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:LONGORDERedset TS2` sets the long ordered set to TS2

`SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:LONGORDERedset?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:LONGORDEREDSET TS1`, indicating the long ordered set is TS1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>

This command sets or queries the PCIE bus search pattern character. The search number is specified by x and can be 1 through 16. The character number is specified by x and can be 1 through 4.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x> <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

Arguments

`<string>` is the pattern character.

Examples

`SEARCH:SEARCH16:TRIGger:A:BUS:PCIE:PATtern:CHAR4 K28.5` sets the pattern character to K28.5.

`SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:PATtern:CHAR1?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:PATTERN:CHAR1 "COM"`, indicating the pattern character is COM.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset

This command sets or queries the PCIE bus search trigger pattern ordered set. The search number is specified by x.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset
{COMPAT|EI|FTS|SKP|INVALID}
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

Arguments

COMPAT is the compliance pattern ordered set COM D21.5 COM D10.2.

EI is the electrical idle ordered set COM IDL IDL IDL.

FTS is the fast training sequence ordered set COM FTS FTS FTS.

SKP is the skip ordered set COM SKP SKP SKP

INVALID is an invalid set. Invalid can be returned, but it should never be sent.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:PATtern:ORDERedset EI sets the ordered set pattern to electrical idle.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:PATtern:ORDERedset? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:PATTERN:ORDEREDSET SKP, indicating the ordered set pattern is skip.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>

This command sets or queries the PCIE bus search trigger pattern symbol minus. The search number is specified by x and can be 1 through 16. Minus is specified by x and can be 1 through 4.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset,
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>,
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR,
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>,
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>,
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus,
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

Arguments

<string> specifies the minus pattern symbol.

Examples

SEARCH:SEARCH15:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus3 “0010000010” sets the minus symbol to 0010000010.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus1? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:PATTERN:SYMBOL:MINUS1 "0011111010", indicating the minus symbol is 0011111010.

SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>

This command sets or queries the PCIE bus search pattern symbol plus. The search number is specified by x and can be 1 through 16. Plus is specified by x and can be 1 through 4.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus, SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

Arguments <string> specifies the plus pattern symbol.

Examples SEARCH:SEARCH11:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS2 “1010110010” sets the plus pattern symbol to 1010110010.

SEARCH:SEARCH1:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS1? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:PCIE:PATTERN:SYMBOL:PLUS1 "1100000101", indicating the plus pattern symbol is 1100000101.

SEARCH:SEARCH<x>:TRIGer:A:BUS:PCIE:SYNCHeader

This command sets or queries the PCIE search sync header. The search number is specified by x. This is only available for data rate 8 Gb/s and above.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGer:A:BUS:PCIE:SYNCHeader
SEARCH:SEARCH<x>:TRIGer:A:BUS:PCIE:SYNCHeader?
```

SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:CONDition

This command sets or queries the search condition for the specified RS232C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:CONDition
{DATa|EOp|PARItyerror|STARt}
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:CONDition?
```

Arguments

`DATa` specifies the search condition as Data.

`EOp` specifies the search condition as End of Packet.

`PARItyerror` specifies the search condition as Parity Error.

`STARt` specifies the search condition as Start.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:CONDITION DATA` sets the search condition for RS232C bus trigger search 1 to Data.

`SEARCH:SEARCH2:TRIGGER:A:BUS:RS232C:CONDITION?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:RS232C:CONDITION EOP`, indicating that the search condition for RS232C bus trigger search 2 is set to End of Packet.

SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:FORMat

This command sets or queries the data format for the specified RS232C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:FORMat?
```

Arguments `BINary` specifies the data format as Binary.

`HEXadecimal` specifies the data format as Hexadecimal.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:DATA:FORMAT BINary` sets the data format for RS232C bus trigger search 1 to BINary.

`SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:DATA:FORMAT?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:DATA:FORMAT HEX`, indicating that the data format for RS232C bus trigger search 1 is set to HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:SIZe

This command sets or queries the number of bits per word in the data string for the specified RS232C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:SIZe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:SIZe?
```

Arguments `<NR3>` is the number of bits per word in the data string, from 1 to 8.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:DATA:SIZE 1` sets the number of bits per word in RS232C bus trigger search 1 data string to 1.

`SEARCH:SEARCH2:TRIGGER:A:BUS:RS232C:DATA:SIZE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:RS232C:DATA:SIZE 8`, indicating that the bits per word for RS232C bus trigger search 2 is set to 8.

SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:VALue

This command sets or queries the data string value for the specified RS232C bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:VALue?
```

Arguments <string> specifies the value of the data string. The valid characters are 0, 1, and X for values in binary format; and A-F, 0-9, and X for values in hexadecimal format.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:DATA:VALUE "01" sets the value of the data string for RS232C bus trigger search 1 to "01" when the format is hexadecimal.

:SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:DATA:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:RS232C:DATA:VALUE "XXXXXXX1", indicating that the data string value for RS232C bus trigger search 1 is set to "XXXXXXX1" when the format is binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:CHAR

Sets or queries the single 8B10B character in the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:CHAR
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:CHAR?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat

Arguments <string> sets the 8B10B character.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CHARACTER:CHAR “K28.5” sets the character in search 1 to K28.5.

SEARCH:SEARCH6:TRIGGER:A:BUS:S8B10B:CHARACTER:CHAR? might return :SEARCH:SEARCH6:TRIGGER:A:BUS:S8B10B:CHARACTER:CHAR “K28.5”, indicating that the character in search 6 is set to K28.5.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus

Sets or queries the specified 8B10B character bit pattern (negative disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:
MINus?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat

Arguments <string> sets the 8B10B character bit pattern.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:MINUS “0011111010” sets the character bit pattern in search 1 to 0011111010.

SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:MINUS? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:MINUS “0011110011”, indicating that the character bit pattern for search 2 is 0011110011

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS

Sets or queries the specified 8B10B character bit pattern (positive disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat

Arguments <string> sets the character bit pattern.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:PLUS "1100000101" sets the character bit pattern in search 1 to 1100000101.

SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:PLUS? might return
:SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:PLUS "1100001100", indicating that the character bit pattern for search 2 is set to 1100001100.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition

This command sets or queries the trigger condition for the specified 8B10B bus trigger search to determine where to place a mark. The search number is represented by <x>.

The selection (character, error, pattern, or any control) must be correct before you can use the corresponding 8B10B search commands (character, error, and pattern).

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition
{ANYControl|CHARacter|ERROR|PATtern}?
```

Related Commands Related commands are shown in the Arguments, below.

Arguments

ANYCONTROL specifies that the oscilloscope will search for any control character.

CHARACTER specifies that the oscilloscope will search for a character, specified by SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:CHAR

ERROR specifies that the oscilloscope will search for errors, specified by SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:CHAR and SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:DISParity.

PATTERN specifies that the oscilloscope will search for a pattern, specified by SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>, or SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CONDITION PATTERN specifies that search 1 will search for a pattern.

SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CONDITION? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CONDITION ERROR, indicating that search 1 will search for errors.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:DISParity

Sets or queries the disparity (positive, negative, or either).

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:DISParity
{NEGAtive|POSITIVe|EITher}
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:DISParity?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:DISParity

Arguments

POSITIVE only the positive values will be used in the search.

NEGATIVE only the negative values will be used in the search.

EITHER positive and negative values will be used in the search.

Examples

SEARCH:SEARCH5:TRIGGER:A:BUS:S8B10B:DISPARITY POSITIVE sets the disparity in search 5 to positive. Only the positive values will be used in the search.

SEARCH:SEARCH5:TRIGGER:A:BUS:S8B10B:DISPARITY? might return SEARCH:SEARCH5:TRIGGER:A:BUS:S8B10B:DISPARITY NEGATIVE, indicating that the disparity in search 5 is set to negative. Only the negative values will be used in the search.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:CHAR

Sets the character error search to on or off, or queries the status of the feature.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:CHAR
{<NR1>|ON|OFF}
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:CHAR?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition

Arguments

<NR1> = 1 enables the character error search. Any other character disables the search.

ON enables the character error search.

OFF disables the character error search.

Examples

SEARCH:SEARCH5:TRIGGER:A:BUS:S8B10B:ERROR:CHAR ON sets search 5 to search for character errors.

SEARCH:SEARCH5:TRIGGER:A:BUS:S8B10B:ERROR:CHAR? might return :SEARCH:SEARCH5:TRIGGER:A:BUS:S8B10B:ERROR:CHAR 0, indicating that search 5 will not search for character errors.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:DISParity

Sets or queries the disparity error search (on or off).

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:DISParity
{<NR1>|ON|OFF}
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:DISParity?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:DISParity

Arguments

<NR1> = 1 enables the disparity error search. Any other character disables the search.

ON enables the disparity error search.

OFF disables the disparity error search.

Examples

SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:ERROR:DISPARITY 1 sets search 2 to search for disparity errors.

SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:ERROR:DISPARITY? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:ERROR:DISPARITY 1, indicating that search 2 will search for disparity errors.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat

Sets the format for the specified search to character or symbol, or queries the state of this feature. Applies to the search character and pattern commands.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat {CHAR|SYMbol}
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:CHAR, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>

Arguments

CHARACTER allows you to edit the values using character format.

SYMBOL allows you to edit the values using symbol format (using the binary equivalent to the character value)

Examples

SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:FORMAT CHARACTER sets the search 2 format to character.

SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:FORMAT? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:S8B10B:FORMAT CHARACTER, indicating that the search 2 format is set to character.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>

Sets or queries the first, second, third or fourth character in the pattern for the specified search.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat

Arguments

<string> is the 8B10B character.

Examples

SEARCH:SEARCH3:TRIGGER:A:BUS:S8B10B:PATTERN:CHAR3 “K28.5” sets the third character in search 3 to K28.5.

SEARCH:SEARCH3:TRIGGER:A:BUS:S8B10B:PATTERN:CHAR4? might return :SEARCH:SEARCH4:TRIGGER:A:BUS:S8B10B:PATTERN:CHAR3 “K28.3”, indicating that the third character in search 4 is set to K28.3.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>

Sets or queries the specified character bit pattern (negative disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:
MINus<x> <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:
MINus<x>?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat

Arguments <string> is the character bit pattern.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:MINUS3 "0011111010" sets the third character bit pattern in search 1 to 0011111010.

SEARCH:SEARCH1:TRIGGGER:A:BUS:S8B10B:PATTERN:SYMBOL:MINUS2? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:MINUS2 "0011110011", indicating that the second character bit pattern in search 1 is set to 0011110011.

SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>

Sets or queries the specified 8B10B character bit pattern (positive disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:
PLUS<x>?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat

Arguments <string> is the 8B10B character bit pattern.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:PLUS3 "1100000101" sets the third character bit pattern in search 1 to 0011111010.

SEARCH:SEARCH1:TRIGGGER:A:BUS:S8B10B:PATTERN:SYMBOL:PLUS2? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:PLUS2 "1100001100", indicating that the second character bit pattern in search 1 is 1100001100.

SEARCH:SEARCH<x>:TRIGger:A:BUS:SOUrce

This command sets or queries the bus source for the specified bus trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:SOUrce
{B0|B1|B2|B3|B4|B5|B6|B7|B8|B9|B10| B11|B12|B13|B14|B15|B16}
SEARCH:SEARCH<x>:TRIGger:A:BUS:SOUrce?
```

Arguments B0–B16 specifies the bus source as a bus number from B01 to B16.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:SOURCE B1 sets the bus source for bus trigger search 1 to B1.

SEARCH:SEARCH1:TRIGGER:A:BUS:SOURCE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:SOURCE B15, indicating that the bus source for bus trigger search 1 is B15.

SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:CONDition

This command sets or queries the trigger condition for the specified SPI bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:CONDition {DATA|SS}
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:CONDition?
```

Arguments

DATA specifies the trigger condition as Data.

SS specifies the trigger condition as Slave Selection.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:CONDITION SS sets the trigger condition for SPI bus trigger search 1 to SS.

SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:CONDITION? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:CONDITION DATA, indicating that the trigger condition for SPI bus trigger search 1 is set to Data.

SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:FORMat

This command sets or queries the data format for the specified SPI bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:FORMat?
```

Arguments

BINary specifies the data format as Binary.

HEXadecimal specifies the data format as Hexadecimal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:DATA:FORMAT BIN sets the data format for SPI bus trigger search 1 to BINary.

SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:DATA:FORMAT? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:DATA:FORMAT HEX, indicating that the data format for SPI bus trigger search 1 is set to HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:SIZe

This command sets or queries the length of the data string in bytes used for the specified SPI bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:SIZe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:SIZe?
```

Arguments <NR3> specifies the number of contiguous data bytes.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:DATA:SIZE 1 sets the length of the data string for SPI bus trigger search 1 to 1 byte.

SEARCH:SEARCH2:TRIGGER:A:BUS:SPI:DATA:SIZE? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:SPI:DATA:SIZE 4, indicating that the length of the data string for SPI bus trigger search 2 is 4 bytes.

SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:VALue

The command sets or queries the data value of the data token for the specified SPI bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:VALue?
```

Arguments <string> specifies the data value in the specified valid format. The valid characters are 0, 1, and X for binary format; and A-F, 0-9, and X for hexadecimal format.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:DATA:VALUE "00001111" sets the data value for SPI bus trigger search 1 to "00001111" in binary format.

SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:DATA:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:SPI:DATA:VALUE "11", indicating that the data value for SPI bus trigger search 1 is set to "11" in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:FORMat

This command sets or queries the address format for normal token for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:FORMat
{BINary|HEX|SYMBolic}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:HIVALue, SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:VALue

Arguments

`BINary` specifies the address format as Binary.

`HEX` specifies the address format as Hexadecimal.

`SYMBolic` specifies the address format as Symbolic.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ADDRESS:FORMAT BINARY` sets the address format for normal token for USB bus trigger search 1 to BINary.

`SEARCH:SEARCH2:TRIGGER:A:BUS:USB:ADDRESS:FORMAT?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:USB:ADDRESS:FORMAT HEX`, indicating that the address format for normal token for USB bus trigger search 2 is set to HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:HIVALue

This command sets or queries the address value for normal token to be used with In Range and Out of Range qualifiers for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

The VALue and HIVALue set a range that the INrange and OUTrange qualifiers use to decide when to trigger. For example, if the QUALIFER is set to INrange, and the address is within the range set by VALue and HIVALue, then a trigger can be generated.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:HIVALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:HIVALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:FORMat, SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:VALue

Arguments <string> specifies the data value in the specified valid format. The valid characters are 0, 1, and X for binary; for hexadecimal; and symbolic).

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ADDRESS:HIVALUE

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ADDRESS:HIVALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ADDRESS:HIVALUE "XX", indicating that the address value for normal token for USB bus trigger search 1 is set to "XX," in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:VALue

This command sets or queries the address value for normal token for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:HIVALue

Arguments <string> specifies the data value in the specified valid format. Valid characters are 0, 1, and X for binary; A-F, 0-9 and X for hexadecimal; and for symbolic.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ADDRESS:VALUE "1110000" sets the address value for the normal token for USB bus trigger search 1 to "1110000" in binary format.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ADDRESS:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ADDRESS:VALUE "11", indicating that the address value for normal token for USB bus trigger search 1 is set to "11" hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:CHAR

Sets or queries the single USB character in the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:CHAR <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:CHAR?
```

Arguments <string> sets the character bit pattern.

Examples SEARCH:SEARCH2:TRIGger:A:BUS:USB:CHARacter:CHAR "0011111010" will search for the 0011111010 character in search 2.

SEARCH:SEARCH2:TRIGger:A:BUS:USB:CHARacter:CHAR? may return "1100000101" indicating the instrument will search for the 1100000101 character in search 2.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus

Sets or queries the specified USB character bit pattern (negative disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus?
```

Arguments <string> sets the character bit pattern.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus "0011111010"` sets the character bit pattern in search 1 to 0011111010.

`SEARCH:SEARCH2:TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:MINUS "0011110011"`, indicating that the character bit pattern for search 2 is 0011110011

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS

Sets or queries the specified character bit pattern (positive disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS?
```

Arguments `<string>` sets the character bit pattern.

Examples `SEARCH:SEARCH1:TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS"1100000101"` sets the character bit pattern in search 1 to 1100000101.

`SEARCH:SEARCH2:TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:PLUS "1100001100"`, indicating that the character bit pattern for search 2 is set to 1100001100.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CONDition

This command sets or queries the search condition for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CONDition
{DATAPacket|EOP|ERROR|HANDSHAKEPacket|RESET|RESUME|
SPECIALPacket|SUSPEND|SYNC|TOKENPacket}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CONDition?
```

Arguments

`DATAPacket` specifies the search condition as Data Packet.

`EOP` specifies the search condition as End of Packet.

`ERROR` specifies the search condition as Error.

`HANDSHAKEPacket` specifies the search condition as Handshake Packet.

`RESET` specifies the search condition as Reset.

`RESUME` specifies the search condition as Resume.

`SPECIALPacket` specifies the search condition as Special Packet.

`SUSPEND` specifies the search condition as Suspend.

`SYNC` specifies the search condition as Sync.

`TOKENPacket` specifies the search condition as Token (Address) Packet.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:CONDITION EOP` sets the search condition for USB bus trigger search 1 to End of Packet.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:CONDITION?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:CONDITION SYNC`, indicating that the search condition for USB bus trigger search 1 is set to Sync.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:FORMat

This command sets or queries the data format for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:FORMat
{BINary|HEX|SYMBolic}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:FORMat?
```

Arguments

`BINary` specifies the data format as Binary.

`HEX` specifies the data format as Hexadecimal.

SYMBolic specifies the data format as Symbolic.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:FORMAT HEX sets the data format for USB bus trigger search 1 to HEXadecimal.

SEARCH:SEARCH2:TRIGGER:A:BUS:USB:DATA:FORMAT? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:USB:DATA:FORMAT BINARY, indicating that the data format for USB bus trigger search 2 is set to BINary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:HIVALue

This command sets or queries the data value for data token used with In Range and Out of Range qualifiers for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:HIVALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:HIVALue?
```

Arguments <string> specifies the data value in the specified valid format (binary, hexadecimal, or symbolic). The valid characters for binary are 0, 1, and X;

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:HIVALUE "XXXXX111" sets the data value for USB bus trigger search 1 to "XXXXX111" in binary format.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:HIVALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:HIVALUE "11", indicating that the data value for USB bus trigger search 1 is set to "11" in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:OFFSet

This command sets or queries the data offset for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:OFFSet
{<NR3>|DONTCare}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:OFFSet?
```

Arguments

<NR3> specifies the data offset in bytes.

DONTCare specifies the data offset as Don't Care.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:OFFSET 5 sets the data offset for USB bus trigger search 1 to 5 bytes.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:OFFSET? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:OFFSET DONTCARE, indicating that the data offset for USB bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:SIZe

This command sets or queries the length of the data string in bytes for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:SIZe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:SIZe?
```

Arguments

<NR3> specifies the data size in bytes.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:SIZE 1 sets the size of the data string for USB bus trigger search 1 to 1 byte.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:SIZE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:SIZE 3, indicating that the size of the data string for USB bus trigger search 1 is set to 3 bytes.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:TYPe

This command sets or queries the data packet type for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:TYPe
{ANY|DATA0|DATA1|DATA2|MDATA}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:TYPe?
```

Arguments `ANY` specifies the data packet type as Any.

`DATA0` specifies the data packet type as DATA0.

`DATA1` specifies the data packet type as DATA1.

`DATA2` specifies the data packet type as DATA2.

`MDATA` specifies the data packet type as MDATA.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:TYPE ANY` sets the data packet type for USB bus trigger search 1 to ANY.

`SEARCH:SEARCH2:TRIGGER:A:BUS:USB:DATA:TYPE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:USB:DATA:TYPE MDADA`, indicating that the packet type for USB bus trigger search 2 is set to MDATA.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:VALue

This command sets or queries the data value for data token for the specified bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:VALue?
```

Arguments `<string>` specifies the data value. The valid characters are 0, 1, and X for binary format; and A-F, 0-9, and X for hexadecimal format.

Examples `SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:VALUE "AB"` sets the data value for data token for USB bus trigger search 1 to "AB" in hexadecimal format.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:DATA:VALUE "XXXXXXXX", indicating that the data value for data token for USB bus trigger search 1 is set to "XXXXXXXX" in binary format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DISParity

Sets or queries the disparity (positive, negative, or either).

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DISParity
{NEGAtive|POSITIVe|EIther}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DISParity?
```

Arguments

POSITIVE only the positive values will be used in the search.

NEGATIVE only the negative values will be used in the search.

EITHER positive and negative values will be used in the search.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:USB:DISParity POSITIVE sets the disparity in search 1 to positive. Only the positive values will be used in the search.

SEARCH:SEARCH5:TRIGger:A:BUS:USB:DISParity? might return SEARCH:SEARCH5:TRIGGER:A:BUS:USB:DISPARITY NEGATIVE, indicating that the disparity in search 5 is set to negative. Only the negative values will be used in the search.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:FORMat

This command sets or queries the endpoint format for data token (to be used with In Range and Out of Range qualifiers) for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:FORMat
{BINary|HEXadecimal|SYMBolic}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:FORMat?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:HIVALue, SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:VALue

Arguments

`BINary` specifies the endpoint format as Binary.

`HEXadecimal` specifies the endpoint format as Hexadecimal.

`SYMBolic` specifies the endpoint format as Symbolic.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:FORMAT BINARY` sets the endpoint format for data token for USB bus trigger search 1 to BINary.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:FORMAT?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:FORMAT HEXADECIMAL`, indicating that the endpoint format for data token for USB bus trigger search 1 is set to HEXadecimal.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:HIVALue

This command sets or queries the endpoint value for data token used with In Range and Out of Range qualifiers for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:HIVALue
<string>}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:HIVALue?
```

Arguments

`<string>` specifies the data value in the specified valid format.

`<No LRN>` specifies do not return a list of instrument settings.

`<No BQ>` specifies do not

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:HIVALUE "0010" sets the upper limit of the endpoint value for data token for USB bus trigger search 1 to "0010" in binary format.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:HIVALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:HIVALUE "X", indicating that the endpoint value for data token for USB bus trigger search 1 is "X" in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:VALue

This command sets or queries the endpoint value for normal token for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:VALue?
```

Arguments <string> specifies the data value in the specified valid format. The valid characters are 0, 1, and X for binary format; A-F, 0-9 and X for hexadecimal format.

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:VALUE "1111" sets the endpoint value for normal token for USB bus trigger search 1 to "1111" in binary format.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ENDPOINT:VALUE "1", indicating that the endpoint value for normal token for USB bus trigger search 1 is set to "1" in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:CHAR

Sets the character error search to on or off, or queries the status of the feature.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:CHAR {<NR1>|OFF|ON}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:CHAR?
```

Arguments <NR1> = 1 enables the character error search. Any other character disables the search.

ON enables the character error search.

OFF disables the character error search.

Examples SEARCH:SEARCH51:TRIGger:A:BUS:USB:ERROR:CHAR ON sets search 5 to search for character errors.

SEARCH:SEARCH5:TRIGger:A:BUS:USB:ERROR:CHAR?might return :SEARCH:SEARCH5:TRIGGER:A:BUS:USB:ERROR:CHAR 0, indicating that search 5 will not search for character errors.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:DISParity

Sets or queries the disparity error search (on or off).

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:DISParity
{<NR1>|OFF|ON}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:DISParity?
```

Arguments <NR1> = 1 enables the disparity error search. Any other character disables the search.

ON enables the disparity error search.

OFF disables the disparity error search.

Examples SEARCH:SEARCH2:TRIGger:A:BUS:USB:ERROR:DISParity1 sets search 2 to search for disparity errors.

SEARCH:SEARCH2:TRIGger:A:BUS:USB:ERROR:DISParity? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:USB:ERROR:DISPARITY 1, indicating that search 2 will search for disparity errors.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:ORDERedset

This command sets or queries the state of searching for an ordered set in the specified search.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:ORDERedset
{|OFF|ON}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:ORDERedset?
```

Arguments OFF disables searching for an ordered set.

ON enables searching for an ordered set.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:USB:ERROR:ORDERedset ON enables searching for an ordered set with search 1.

SEARCH:SEARCH1:TRIGger:A:BUS:USB:ERROR:ORDERedset? may return OFF indicating that searching for an ordered set using search 1 is disabled.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERRTYPE

This command sets or queries the error type for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERRTYPE
{BITSTUFFing|CRC5|CRC16|PID}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERRTYPE?
```

Arguments BITSTUFFing specifies the error type as Bit Stuffing.

CRC5 specifies the error type as Token CRC5 (Cyclic Redundancy Check 5).

CRC16 specifies the error type as Data CRC16 (Cyclic Redundancy Check 16).

PID specifies the error type as PID Check Bits.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ERRTYPE CRC5 sets the error type for USB bus trigger search 1 to Token CRC5.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ERRTYPE? might return SEARCH:SEARCH1:TRIGGER:A:BUS:USB:ERRTYPE PID, indicating that the error type for USB bus trigger search 1 is set to PID Check Bits.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:FORMat

Sets the format for the specified search to character or symbol, or queries the state of this feature. Applies to the search character and pattern commands.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:FORMat {CHAR|SYMbol}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:FORMat?
```

Arguments

CHARACTER allows you to edit the values using character format.

SYMBOL allows you to edit the values using symbol format (using the binary equivalent to the character value).

Examples

SEARCH:SEARCH2:TRIGger:A:BUS:USB:FORMatCHARACTER sets the search 2 format to character.

SEARCH:SEARCH2:TRIGger:A:BUS:USB:FORMat? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:USB:FORMAT CHARACTER, indicating that the search 2 format is set to character.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:HANDSHAKEType

This command sets or queries the handshake type for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:HANDSHAKEType
{ACK|ANY|NAK|NYET|STALL}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:HANDSHAKEType?
```

Arguments

`ACK` specifies the handshake type as Acknowledgement (positive) (XX10).

`ANY` specifies the handshake type as Any (0010).

`NAK` specifies the handshake type as Negative Acknowledgment (1010).

`NYET` specifies the handshake type as No response Yet (0110).

`STALL` specifies the handshake type as Stall (endpoint is halted or control pipe request not supported) (1110).

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:HANDSHAKETYPE NAK` sets the handshake type for USB bus trigger search 1 to NAK.

`SEARCH:SEARCH2:TRIGGER:A:BUS:USB:HANDSHAKETYPE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:USB:HANDSHAKETYPE NYET`, indicating that the handshake type for USB bus trigger search 2 is set to NYET.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:LONGORDERedset

This command sets or queries the USB bus search trigger long ordered set. The search number is specified by x.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:LONGORDERedset
{EIE|TS1|TS2}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:LONGORDERedset?
```

Arguments

`EIE` is the Electrical Idle Exit ordered set.

`TS1` is the Training Sequence 1 ordered set.

`TS2` is the Training Sequence 2 ordered set.

Examples

`SEARCH:SEARCH1:TRIGger:A:BUS:USB:LONGORDERedset TS2` sets the long ordered set to TS2

SEARCH:SEARCH1:TRIGger:A:BUS:USB:LONGORDERedset? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:LONGORDEREDSET TS1, indicating the long ordered set is TS1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets

Sets or queries the USB bus search packets to mark. This setting only applies when search type is set to Packets. This setting is only available when the USB bitrate is set to SS.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets {LMP | TP | DP
| ITP}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets?
```

Arguments

LMP mark LMP packets,

TP mark TP packets,

DP mark DP packets,

ITP Mmark ITP packets.

Examples

```
SEARCH:SEARCH1:TRIGger:A:BUS:USB:PACKets LMP marks LMP
packets.
```

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PACKets? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:PACKETS LMP

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:LMPPACKets

Sets or queries the LMP packets to mark when system is searching for LMP packets. This setting is only available when the USB bitrate is set to SS.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:LMPPACKets {ANY |
LMPLink | LMPUtwo | LMPDevice | LMPConfig | LMPResponse}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:LMPPACKets?
```

Arguments

ANY mark all LMP packets

LMPLink mark only link packets

LMPUtwo mark only U2 packets

LMPDevice mark only Device packets

LMPConfig mark only Config Packets

LMPResponse mark only response packets

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PACKets ANY marks all LMP packets.

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PACKets? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:PACKETS:LMPPACKETS ANY..

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:TPPACKets

Sets or queries the TP packets to mark when the system is searching for TP packets. This setting is only available when the USB bitrate is set to SS.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:TPPACKets {ANY
| TPACK | TPNRDY | TPERDY | TPStatus | TPSTall | TPNotify
| TPPing | TPResponse}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:TPPACKets?
```

Arguments

ANY mark all TP packets

TPACK mark all ACK packets

TPNRDY mark al NRDY (not ready) packets

TPERDY mark all ERDY packets.

TPStatus mark all status packets.

TPSTall mark all stall packets.

TPNotify mark all notify packets.

TPPing mark all ping packets.

TPResponse mark all response packets.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:USB:PACKets:TPPACKets TPSTall marks all stall packets.

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PACKets:TPPACKets? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:PACKETS:TPPACKETS ANY.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:CHAR<x>

Sets or queries the first, second, third or fourth character in the pattern for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:CHAR<x> <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:CHAR<x>?
```

Arguments <string> is the character.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:CHAR1 "0011111010" searches for 0011111010 as the first character in search 1.

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:CHAR1?might return "11000001010" as the first character in search 1.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:NUMSymbols

Sets or queries the number of symbols to search for when USB search type is in pattern mode. This setting is only available when the USB bitrate is set to SS.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:NUMSymbols <NR1>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:NUMSymbols?
```

Arguments <NR1> is the number 2, 3, or 4.

Examples SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:NUMSymbols 2 sets the number of symbols to 2.

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:NUMSymbols? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:PATTERN:NUMSYMBOLS 2.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:ORDERedset

This command sets or queries the state of searching for an ordered set.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:ORDERedset
{OFF|ON|0|1}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:ORDERedset?
```

Arguments

OFF disables searching for an ordered set.

ON enables searching for an ordered set.

0 disables searching for an ordered set.

1 enables searching for an ordered set.

Examples

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:ORDERedset ON enables searching for an ordered set on search 1.

SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:ORDERedset? may return OFF indicating that searching for an ordered set on search 1 is disabled.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>

Sets or queries the specified character bit pattern (negative disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>?
```

Arguments

<string> is the character bit pattern.

Examples

`SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus3 "0011111010"` sets the third character bit pattern in search 1 to 0011111010.

`SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus2?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:MINUS2 "0011110011"`, indicating that the second character bit pattern in search 1 is set to 0011110011.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>

Sets or queries the specified character bit pattern (positive disparity) for the specified search.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>
<string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>?
```

Arguments <string> is the symbol bit pattern.

Examples

`SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus3 "1100000101"` sets the third character bit pattern in search 1 to 0011111010.

`SEARCH:SEARCH1:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus2?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:PLUS2 "1100001100"`, indicating that the second character bit pattern in search 1 is 1100001100.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:QUAlifier

This command sets or queries the qualifier for address and data for the specified bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:QUAlifier
{EQUal|INrange|LESSEQual|MOREEQual|
OUTrange|UNEQual|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:QUAlifier?
```

Arguments

`EQUal` specifies the qualifier as Equal.

`INrange` specifies the qualifier as Inside Range.

`LESSEQual` specifies the qualifier as Less Than or Equal to.

`MOREEQual` specifies the qualifier as More Than or Equal to.

`OUTrange` specifies the qualifier as Out of Range.

`UNEQual` specifies the qualifier as Unequal.

`LESSThan` specifies the qualifier as Less Than.

`MOREThan` specifies the qualifier as More Than.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:QUALIFIER EQUAL` sets the qualifier for USB bus trigger search 1 to EQUal.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:QUALIFIER?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:QUALIFIER UNEQUAL`, indicating that the qualifier for USB bus trigger search 1 is set to UNEQual.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FORMat

This command sets or queries the format for the Start of Frame frame number for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FORMat
{BINary|HEXadecimal|SYMBolic}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FRAMENUMber

Arguments

`BINary` specifies the format as Binary.

`HEXadecimal` specifies the format as Hexadecimal.

`SYMBolic` specifies the format as Symbolic.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SOF:FORMAT HEX` sets the format for the SOF frame number for USB bus trigger search 1 to HEXadecimal.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SOF:FORMAT?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SOF:FORMAT BINARY`, indicating that the format for the SOF frame number for USB bus trigger search 1 is set to BINary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FRAMENUMber

This command sets or queries the frame number string to use for the Start of Frame for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FRAMENUMber <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FRAMENUMber?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FORMat

Arguments

`<string>` specifies the frame number string for the Start of Frame in a valid format (binary, hexadecimal, or symbolic).

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SOF:FRAMENUMBER "1001"` sets the frame number string for the SOF for USB bus trigger search 1 to "1001" in symbolic format.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SOF:FRAMENUMBER?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SOF:FRAMENUMBER "xxxxxxx1001"`, indicating that the frame number string for the SOF for USB bus trigger search 1 is set to "XXXXXXX1001" in binary format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPECIALType

This command sets or queries the PID value for the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPECIALType
{ANY|ERR|PING|PRE|RESERVED|SPLIT}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPECIALType?
```

Arguments

`ANY` specifies the PID value as Any (XX00).

`ERR` specifies the PID value as ERR (1100).

`PING` specifies the PID value as PING (0100).

`PRE` specifies the PID value as PRE (1100).

`RESERVED` specifies the PID value as Reserved (0000).

`SPLIT` specifies the PID value as Split (1000).

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPECIALTYPE ERR` sets the PID value for USB bus trigger search 1 to ERR.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPECIALTYPE?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPECIALTYPE SPLIT`, indicating that the PID value for USB bus trigger search 1 is set to SPLIT.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:ET:VALue

This command sets or queries the Endpoint Type value for the specified USB bus trigger search on split token field to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:ET:VALue
{BULK|CONTROL|DONTCare|INTERRUPT|ISOchronous}
```

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:ET:VALue?

Arguments

BULK specifies the Endpoint Type value as Bulk (10).

CONTROL specifies the Endpoint Type value as Control (00).

DONTCare specifies the Endpoint Type value as Don't Care.

INTERRUPT specifies the Endpoint Type value as Interrupt (11).

ISOchronous specifies the Endpoint Type value as Isochronous (01).

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:ET:VALUE BULK sets the Endpoint Type value for USB bus trigger search 1 to Bulk.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:ET:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:ET:VALUE DONTCARE, indicating that the Endpoint Type value for USB bus trigger search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:FORMat

This command sets or queries the format for the hub address of the specified USB bus trigger search on split token field to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:VALue

Arguments

BINary specifies the hub address format as Binary.

HEXadecimal specifies the hub address format as Hexadecimal.

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:HUB:FORMAT HEX sets the hub address format for USB bus trigger search 1 to HEXadecimal.

SEARCH:SEARCH2:TRIGGER:A:BUS:USB:SPLIT:HUB:FORMAT? might return :SEARCH:SEARCH2:TRIGGER:A:BUS:USB:SPLIT:HUB:FORMAT BINARY, indicating that the hub address format for USB bus trigger search 2 is set to BINary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:VALue

This command sets or queries the hub address of the specified USB bus trigger on split token field search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:VALue?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:FORMat

Arguments <string> specifies the hub address. The valid characters are .

Examples SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:HUB:VALUE "1001" sets the hub address to "XXX1001".

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:HUB:VALUE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:HUB:VALUE "XX" (for binary format), indicating that the hub address for the USB bus trigger search on split token field is "XX".

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:FORMat

This command sets or queries the port address format for the specified USB bus trigger on split token field search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:FORMat
{BINary|HEXadecimal}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:FORMat?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:VALue

Arguments

`BINary` specifies the port address format as Binary.

`HEXadecicmal` specifies the port address format as Hexadecimal.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:PORT:FORMAT HEX` sets the port address format for USB bus trigger search 1 to hexadecimal.

`SEARCH:SEARCH2:TRIGGER:A:BUS:USB:SPLIT:PORT:FORMAT?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:USB:SPLIT:PORT:FORMAT BINARY`, indicating that the port address format for USB bus trigger search 2 on split token field is set to binary.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:VALue

This command sets or queries the port address for the specified USB bus trigger on split token field search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:VALue <string>
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:VALue?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:FORMat

Arguments

`<string>` specifies the port address in the valid format. The valid characters are

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:PORT:VALUE "1111"` sets the port address for USB bus trigger search 1 to "XXX1111" for binary format.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:PORT:VALUE?` might return `SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:PORT:VALUE "XX"`,

indicating that the port address for USB hub trigger search 1 is set to “XX” in hexadecimal format.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SC:VALue

This command sets or queries the Start/Complete value for the specified USB bus trigger on split token field search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SC:VALue
{CSPLIT|DONTCare|SSPLIT}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SC:VALue?
```

Arguments

`CSPLIT` specifies Complete (CSPLIT)(1) Start/Complete value.

`DONTCare` specifies the Start/Complete value as Don’t Care (X).

`SSPLIT` specifies Start (SSPLIT) (0) Start/Complete value.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:SC:VALUE DONTCARE` sets the Start/Complete value for USB bus trigger search 1 to DONTCare.

`SEARCH:SEARCH2:TRIGGER:A:BUS:USB:SPLIT:SC:VALUE?` might return `:SEARCH:SEARCH2:TRIGGER:A:BUS:USB:SPLIT:SC:VALUE SSPLIT`, indicating that the Start/Complete value for USB bus trigger search 2 is set to SSPLIT.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SE:VALue

This command sets or queries the Start/End value for the specified USB bus trigger on split token field search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SE:VALue
{DONTCare|FULLSPeed|ISOALL|ISOEND|ISOMID|ISOSTART|LOWSPeed}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SE:VALue?
```

Arguments

`DONTCare` specifies Don't Care (X) Start/End bit value.

`FULLSPeed` specifies Control/Bulk/Interrupt Full Speed device (0X) Start/End bit value.

`ISOALL` specifies Isochronous Data is All (11) Start/End bit value.

`ISOEND` specifies Isochronous Data is End (01) Start/End bit value.

`ISOMID` specifies Isochronous Data is Middle (00) Start/End bit value.

`ISOSTART` specifies Isochronous Data is Start (10) Start/End bit value.

`LOWSPeed` specifies Control/Bulk/Interrupt Low Speed device (1X) Start/End bit value.

Examples

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:SE:VALUE ISOSTART` sets the Start/End value for USB bus trigger on split token field search 1 to ISOSTART.

`SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:SE:VALUE?` might return `:SEARCH:SEARCH1:TRIGGER:A:BUS:USB:SPLIT:SE:VALUE DONTCARE`, indicating that the Start/End value for USB bus trigger on split token field search 1 is set to DONTCare.

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:TOKENType

This command sets or queries the token type when the specified USB bus trigger search to determine where to place a mark. <x> is the search number.

NOTE. *Not available on some models.*

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:TOKENType
{ANY|IN|OUT|SETUP|SOF}
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:TOKENType?
```

Arguments

`ANY` specifies the token type as Any (XX01).

`IN` specifies the token type as IN (1001).

`OUT` specifies the token type as OUT (0001).

SETUP specifies the token type as SETUP (1101).

SOF specifies the token type as Start Of Frame (0101).

Examples

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:TOKENTYPE IN sets the token type for USB bus trigger search 1 to IN.

SEARCH:SEARCH1:TRIGGER:A:BUS:USB:TOKENTYPE? might return :SEARCH:SEARCH1:TRIGGER:A:BUS:USB:TOKENTYPE SOF, indicating that the token type for USB bus trigger search 1 is set to Start Of Frame.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOBitrate

This command sets or queries the auto bit rate value for a DDR Memory analysis search to determine where to place a mark. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOBitrate
{AUTO|CUSTom|DDR200|DDR266|DDR333|DDR370|
DDR400|DDR533|DDR667|DDR800|DDR1066|DDR1200|DDR1333|DDR1466|
DDR1600|DDR1866|DDR2133|DDR2400|DDR2666|DDR3200|DDR4000|
DDR4800|DDR5000|DDR5500}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOBitrate?
```

Arguments

AUTO sets the auto bit rate to auto. When auto is selected, the instrument determines the bit rate from the applied signal.

CUSTom sets the auto bit rate to custom. When custom is selected, you set the bit rate value.

DDR200 – DDR5500 sets the bit rate appropriate for the specified standard.

Examples

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:AUTOBITRATE CUSTOM sets the auto bit rate to custom and you can set the desired bit rate value.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:AUTOBITRATE? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:AUTOBITRATE AUTO, indicating that the auto bit rate is set to auto.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel

This command sets or queries the reference levels for a DDR Memory analysis search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel
{AUTO|CUSTom}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel?
```

Arguments AUTO sets the reference levels to auto. When auto is selected, the instrument determines the reference level from the applied signal.

CUSTom sets the reference levels to custom. When custom is selected, you set the reference level

Examples SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:AUTOREFLEVEL CUSTOM sets the DDR reference levels to custom and you can set the desired reference level.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:AUTOREFLEVEL? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:AUTOREFLEVEL AUTO, indicating that the reference levels are set to auto.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:BITRate

This command sets or queries the data rate value for a DDR Memory analysis search to determine where to place a mark. You can set the bit rate only when AUTOBITRATE is Custom. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:BITRate <NR3>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:BITRate?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOBitRate

Arguments <NR3> specifies the DDR data rate in bits per second.

Examples SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:BITRATE 8.0 +36 sets the DDR data rate to 8.0E+30 Mb/s.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:BITRATE? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:BITRATE 9.9000E+36, indicating that the DDR data rate is set to 9.9 E+30 Mb/s.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSActive

This command sets or queries whether the expected chip-select will be an active HIGH or LOW.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSActive {HIGH|LOW}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSActive?
```

Arguments

`HIGH` sets the search to look for an active HIGH.

`LOW` sets the search to look for an active HIGH.

Examples

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSACTIVE HIGH` sets the DDR memory search chip select to an active HIGH.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSACTIVE?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSACTIVE LOW`, indicating the chip select is set to an active LOW.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSLevel

This command sets or queries the DDR memory search chip-select level.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSLevel <NR3>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSLevel?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSMode

Arguments

`<NR3>` is the amount of chip-select level in volts.

Examples

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSLEVEL 1.1` sets the chip-select level for DDR memory searches to 1.1 volts.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSLEVEL?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSLEVEL 0.0000`, indicating the chip-select level for DDR memory searches is set to zero volts.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSMode

This command sets or queries the chip-select mode for DDR memory searches.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSMode {AUTO|MANual}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSMode?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSLevel

Arguments

AUTO mode calculates the chip-select level for you.

MANUAL lets you manually set the chip-select level.

Examples

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSMODE MANUAL sets the chip-select mode for DDR memory searches to MANUAL.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSMODE? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSMODE AUTO, indicating the chip-select mode for DDR memory searches is set to AUTO.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSSource

This command sets or queries the chip select source for DDR memory searches.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSSource
{CH<x>|MATH<x>|REF<x>|NONe}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSSource?
```

Arguments

CH<x> specifies one input channel as the search source, where <x> is the channel number and can be 1, 2, 3, or 4.

MATH<x> specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

REF<x> specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

NONe specifies no CSSource.

Examples `SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSSOURCE REF1` sets the chip select source for DDR memory searches to reference 1.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSSOURCE?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:CSSOURCE CH1`, indicating the chip select source for DDR memory searches is channel 1.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:DATASource

This command sets or queries the data source for a DDR Memory analysis search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:DATASource
{CH<x>|MATH<x>|REF<x>}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:DATASource?
```

Arguments `CH<x>` specifies one input channel as the search source, where <x> is the channel number.

`MATH<x>` specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

`REF<x>` specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

Examples `SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:DATASOURCE CH2` sets the DDR data source to CH2.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:DATASOURCE?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:DATASOURCE CH1`, indicating that the DDR data source is set to CH1.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:HYSteresis

This command sets or queries the hysteresis percent for DDR memory searches.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:HYSteresis <NR3>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:HYSteresis?
```

Arguments <NR3> is the hysteresis amount in percent.

Examples SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:HYSTERESIS 0.10 sets the hysteresis for DDR memory searches to 0.10 percent.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:HYSTERESIS? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:HYSTERESIS 1.0000, indicating the hysteresis for DDR memory searches is set to 1.0 percent.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MARgin

This command sets or queries the termination logic margin for back-to-back writes.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MARgin <NR3>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MARgin?
```

Arguments <NR3> is the termination logic margin in percent.

Examples SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MARGIN 1.0000 sets the termination logic margin to 1.0 percent.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MARGIN? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MARGIN 0.0000, indicating the termination logic margin is set to 0.0 percent.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MAXCas

This command sets or queries the maximum distance in clock periods, to the left of the start of the mark where the instrument looks for the chip-select pulse.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MAXCas <NR3>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MAXCas?
```

Arguments <NR3> is the maximum distance in clock periods, to the left of the start of the mark where the instrument looks for the chip-select pulse.

Examples

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MAXCAS 2.0000` sets `MAXCas` to 2.0 clock periods to the left of the start of the mark.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MAXCAS?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MAXCAS 3.0000`, indicating `MAXCas` is set to 3.0 clock periods to the left of the mark.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MINCas

This command sets or queries the minimum distance in clock periods, to the left of the start of the mark where the instrument looks for the chip-select pulse.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MINCas <NR3>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MINCas?
```

Arguments

`<NR3>` is the minimum distance in clock periods, to the left of the start of the mark where the instrument looks for the chip-select pulse.

Examples

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MINCAS 2.0000` sets `MINCas` to 2.0 clock periods to the left of the start of the mark.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MINCAS?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:MINCAS 2.0000`, indicating `MINCas` is set to 2.0 clock periods to the left of the mark.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:HIGH

This command sets or queries the value of the data HIGH reference level for a DDR Memory analysis search to determine where to place a mark. You can set the value only when AUTOREFLEVEL is Custom. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:HIGH
<NRF>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:HIGH?
```

Related Commands

`SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel`

Arguments <NRF> specifies the value of the data HIGH reference level in volts.

Examples SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:HIGH 1 sets the data HIGH reference level to 1.0 volt.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:HIGH? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:HIGH 0.0000, indicating that the data HIGH reference level is set to 0 volt.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:LOW

This command sets or queries the value of the data LOW reference level for a DDR Memory analysis search to determine where to place a mark. You can set the value only when AUTOREFLEVEL is Custom. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:LOW <NRF>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:LOW?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel

Arguments <NRF> specifies the value of the data LOW reference level in volts.

Examples SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:LOW 0.2 sets the data LOW reference level to 0.2 volt.

SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:LOW? might return :SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:LOW 200.0000E-3, indicating that the data LOW reference level is set to 0.2 volt.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:MID

This command sets or queries the value of the data MID reference level for a DDR Memory analysis search to determine where to place a mark. You can set the value only when AUTOREFLEVEL is Custom. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:MID <NRF>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:MID?
```

Related Commands

`SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel`

Arguments

`<NRF>` specifies the value of the data MID reference level in volts.

Examples

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:MID 1.5` sets the data MID reference level to 1.5 volts.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:MID?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:DATA:MID 1.0000`, indicating that the data MID reference level is set to 1.0 volt.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:HIGH

This command sets or queries the value of the strobe HIGH reference level for a DDR Memory analysis search to determine where to place a mark. You can set the value only when AUTOREFLEVEL is Custom. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:HIGH
<NRF>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:HIGH?
```

Related Commands

`SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel`

Arguments

`<NRF>` specifies the value of the strobe HIGH reference level in volts.

Examples

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:HIGH 5.0` sets the strobe HIGH reference level to 5.0 volts.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:HIGH?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:HIGH 0.000`, indicating that the strobe HIGH reference level is set to 0 volt.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:LOW

This command sets or queries the value of the strobe LOW reference level for a DDR Memory analysis search to determine where to place a mark. You can set the value only when AUTOREFLEVEL is Custom. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:LOW
<NRF>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:LOW?
```

Related Commands `SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel`

Arguments `<NRF>` specifies the value of the strobe LOW reference level in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:LOW 1` sets the strobe LOW reference level to 1.0 volt.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:LOW?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:LOW 0.0000`, indicating that the strobe LOW reference level is set to 0 volt.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:MID

This command sets or queries the value of the strobe MID reference level for a DDR Memory analysis search to determine where to place a mark. You can set the value only when AUTOREFLEVEL is Custom. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:MID
<NRF>
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:MID?
```

Related Commands `SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel`

Arguments `<NRF>` specifies the value of the strobe MID reference level in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:MID 3` sets the strobe MID reference level to 3.0 volts.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:MID?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:REFLEVEL:STROBE:MID 0.0000`, indicating that the strobe MID reference level is set to 0 volt.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STANdard

This command sets or queries the DDR standard for a DDR Memory analysis search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STANdard
{DDR|DDR2|DDR3|GDDR3|LPDDR|DDR3L|DDR4|LPDDR3}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STANdard?
```

Arguments

`<DDR>` sets the DDR standard type to DDR.

`<DDR2>` sets the DDR standard type to DDR2.

`<DDR3>` sets the DDR standard type to DDR3.

`<GDDR3>` sets the DDR standard type to GDDR3.

`<LPDDR>` sets the DDR standard type to LPDDR.

`<DDR3L>` sets the DDR standard type to DDR3L.

`<DDR4>` sets the DDR standard type to DDR4.

`<LPDDR3>` sets the DDR standard type to LPDDR3.

Examples `SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:STANDARD DDR2` sets the DDR standard to DDR2.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:STANDARD?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:STANDARD DDR3`, indicating that the DDR standard is set to DDR3.

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STROBEsource

This command sets or queries the strobe source for a DDR Memory analysis search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STROBEsource
{CH<x>|MATH<x>|REF<x>}
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STROBEsource?
```

Arguments `CH<x>` specifies the input channel as the search source, where <x> is the channel number.

`MATH<x>` specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

`REF<x>` specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

Examples `SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:STROBESOURCE REF2` sets the DDR strobe source to REF2.

`SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:STROBESOURCE?` might return `:SEARCH:SEARCH1:TRIGGER:A:DDRMEMORY:STROBESOURCE CH1`, indicating that the DDR strobe source is set to CH1.

SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:CH<x>

This command sets or queries the slope for an edge trigger search to determine where to place a mark. <x> is the search number or the channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:CH<x>
{RISe|FALL|EITher}
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:CH<x>?
```

Arguments `RISe` specifies a rising edge.

`FALL` specifies a falling edge.

`EITher` specifies either rising or falling edge.

Examples `SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:CH2 RISE` sets the Channel 2 slope for search 1 to rise.

SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:CH1 RISE, indicating that the slope for the Channel 1 trigger for search 1 is rise.

SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:MATH<x>

This command sets or queries the slope for an edge trigger search to determine where to place a mark. <x> is the search number or the math waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:MATH<x>
{RISe|FALL|EITher}
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:MATH<x>?
```

Arguments RISe specifies a rising edge.

FALL specifies a falling edge.

EITHER specifies either rising or falling edge.

Examples SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:MATH2 RISE sets the Math 2 waveform slope for search 1 to rise.

SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:MATH1 RISE, indicating that the slope for the Math 1 waveform trigger for search 1 is rise.

SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:REF<x>

This command sets or queries the slope for an edge trigger search to determine where to place a mark. <x> is the search number or the reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:REF<x>
{RISe|FALL|EITher}
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:REF<x>?
```

Arguments RISe specifies a rising edge.

FALL specifies a falling edge.

EITHER specifies either rising or falling edge.

Examples

SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:REF2 RISE sets the Reference 2 waveform slope for search 1 to rise.

SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:EDGE:SLOPE:REF1 FALL, indicating that the slope for the Reference 1 waveform trigger for search 1 is the falling edge.

SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOUrce

This command sets or queries the source waveform for an edge trigger search to determine where to place a mark. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOUrce
{CH<x>|MATH<x>|REF<x>}
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<x> specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

REF<x> specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

Examples

SEARCH:SEARCH1:TRIGGER:A:EDGE:SOURCE CH2 sets the source waveform for the A trigger to Channel 2.

SEARCH:SEARCH1:TRIGGER:A:EDGE:SOURCE? might return :SEARCH:SEARCH1:TRIGGER:A:EDGE:SOURCE CH2, indicating that the Channel 2 edge trigger is the source for search 1.

SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:CH<x>

This command sets or queries the polarity for a glitch trigger search to determine where to place a mark. <x> is the search number or channel number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:CH<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:CH<x>?
```

Arguments

`EITher` specifies positive or negative polarity.

`NEGAtive` specifies a negative polarity.

`POSITIVe` specifies a positive glitch polarity.

Examples

`SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:CH2 NEGATIVE` specifies that the Channel 2 glitch polarity must be negative for search 1.

`SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:CH1 NEGATIVE`, indicating that the Channel 1 glitch polarity for search 1 is negative.

SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:MATH<x>

This command sets or queries the polarity for a glitch trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:MATH<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:MATH<x>?
```

Arguments

`EITher` specifies positive or negative polarity.

`NEGAtive` specifies a negative polarity.

`POSITIVe` specifies a positive glitch polarity.

Examples

`SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:MATH2 NEGATIVE` specifies that the Math 2 waveform glitch polarity must be negative for search 1.

`SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:MATH1 POSITIVE`, indicating that the Math 1 waveform glitch polarity for search 1 is positive.

SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:REF<x>

This command sets or queries the polarity for a glitch trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:REF<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:REF<x>?
```

Arguments

`EITher` specifies positive or negative polarity.

`NEGAtive` specifies a negative polarity.

`POSITIVe` specifies a positive glitch polarity.

Examples

`SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:REF2 NEGATIVE` specifies that the Reference 2 waveform glitch polarity must be negative for search 1.

`SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:GLITCH:POLARITY:REF1 POSITIVE`, indicating that the Reference 1 waveform glitch polarity for search 1 is positive.

SEARCH:SEARCH<x>:TRIGger:A:GLItch:TRIGIF

This command sets or queries the accept or reject state for a glitch trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:GLItch:TRIGIF {ACCept|REJect}
SEARCH:SEARCH<x>:TRIGger:A:GLItch:TRIGIF?
```

Arguments

`ACCept` specifies that a glitch occurs.

`REJect` specifies that a glitch does not occur.

Examples

`SEARCH:SEARCH1:TRIGGER:A:GLITCH:TRIGIF ACCEPT` specifies to place a mark if the A trigger glitch occurs.

SEARCH:SEARCH1:TRIGGER:A:GLITCH:TRIGIF? might return :SEARCH:SEARCH1:TRIGGER:A:GLITCH:TRIGIF ACCEPT, indicating that a mark will be placed if a glitch occurs.

SEARCH:SEARCH<x>:TRIGger:A:GLItch:WIDth

This command sets or queries the glitch width for a glitch trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:GLItch:WIDth <NR3>
SEARCH:SEARCH<x>:TRIGger:A:GLItch:WIDth?
```

Arguments <NR3> specifies the glitch width in seconds.

Examples SEARCH:SEARCH1:TRIGGER:A:GLITCH:WIDTH 0.000000002 sets the search1 glitch width to 2.0 ns.

SEARCH:SEARCH1:TRIGGER:A:GLITCH:WIDTH? might return :SEARCH:SEARCH1:TRIGGER:A:GLITCH:WIDTH 1.50E-6, indicating that the search 1 glitch width is set to 1.50 µs.

SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x>

This command sets or queries the level for an edge trigger search to determine where to place a mark. <x> is the search number or the channel number. Each channel can have an independent level.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x> {<NR3>|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x>?
```

Arguments <NR3> specifies the trigger level in volts.

ECL specifies a -1.3 V trigger level.

TTL specifies a 1.4 V trigger level.

Examples `SEARCH:SEARCH1:TRIGGER:A:LEVEL:CH2 1.0` sets the Channel 1 A trigger level for search 1 to 1.0 Volts.

`SEARCH:SEARCH1:TRIGGER:A:LEVEL:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:LEVEL:CH1 200.0000E-3`, indicating the Channel 1 trigger level for search 1 is 200 mV.

SEARCH:SEARCH<x>:TRIGger:A:LEVel:MATH<x>

This command sets or queries the math waveform level for an edge trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:LEVel:MATH<x> {<NR3>|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:LEVel:MATH<x>?
```

Arguments `<NR3>` specifies the trigger level in volts.

`ECL` specifies a -1.3 V trigger level.

`TTL` specifies a 1.4 V trigger level.

Examples `SEARCH:SEARCH1:TRIGGER:A:LEVEL:MATH2 0.20` sets the Math 2 waveform trigger level for search 1 to 200 mV.

`SEARCH:SEARCH1:TRIGGER:A:LEVEL:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:LEVEL:MATH1 100.0000E-3`, indicating that the Math 2 waveform trigger level for search 1 is set to 100 mV.

SEARCH:SEARCH<x>:TRIGger:A:LEVel:REF<x>

This command sets or queries the specified reference waveform level for a 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 and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:LEVel:REF<x> {<NR3>|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:LEVel:REF<x>?
```

Arguments <NR3> specifies the trigger level in volts.

ECL specifies a -1.3 V trigger level.

TTL specifies a 1.4 V trigger level.

Examples SEARCH:SEARCH1:TRIGGER:A:LEVEL:REF2 0.20 sets the Reference 2 waveform trigger level for search 1 to 200 mV.

SEARCH:SEARCH1:TRIGGER:A:LEVEL:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:LEVEL:REF1 200.0000E-3, indicating that the Reference 1 waveform trigger level for search 1 is set to 200 mV.

SEARCH:SEARCH<x>:TRIGger:A:LOGIc:FUNCtion

This command sets or queries the logic operator for a pattern or state trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:FUNCtion {AND|NANd|NOR|OR}
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 are false.

NOR places a mark if all conditions are false.

OR places a mark if any of the conditions are true.

Examples SEARCH:SEARCH1:TRIGGER:A:LOGIC:FUNCTION AND sets the trigger a logic function for search 1 to AND.

SEARCH:SEARCH1:TRIGGER:A:LOGIC:FUNCTION? might return :SEARCH:SEARCH1:TRIGGER:A:LOGIC:FUNCTION NOR, indicating that the logic function for search 1 is set to NOR.

SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:CH<x>

This command sets or queries the Boolean logic criteria for a pattern or state trigger search to determine where to place a mark. <x> is the search number or the channel number.

Group Search and Mark

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.

Examples `SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:CH2 HIGH` sets the logic input for Channel 1 to HIGH.

`SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:CH1 LOW`, indicating the logic input on Channel 1 for search 1 is set to LOW.

SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:MATH<x>

This command sets or queries the Boolean logic criteria for a pattern or state trigger search to determine where to place a mark. <x> is the search number or the math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:MATH<x> {HIGH|LOW|X}
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:MATH<x>?
```

Arguments `HIGH` specifies a high logic level.

`LOW` specifies a low logic level.

`X` specifies a “don't care” condition.

Examples `SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:MATH2 HIGH` sets the Math 2 waveform logic input for search 1 to HIGH.

`SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:MATH1 LOW`, indicating the Math 1 waveform logic input for search 1 is set to LOW.

SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:REF<x>

This command sets or queries the Boolean logic criteria for a pattern or state trigger search to determine where to place a mark. <x> is the search number or the reference channel number.

Group Search and Mark

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.

Examples

SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:REF2 HIGH sets the Reference 2 waveform logic input for search 1 to HIGH.

SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:LOGIC:INPUT:REF1 LOW, indicating the Reference 1 waveform login input for search 1 is set to LOW.

SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x>

This command sets or queries the channel threshold level for a pattern or state trigger search to determine where to place a mark. SEARCH<x> is the search number and CH<x> is the channel number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x>?
```

Arguments

<NR3> specifies the threshold level in volts.

Examples

SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:CH2 1.3 sets the Channel 2 logic threshold for search 1 to 1.3 Volts.

SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:CH1 2.0000, indicating the Channel 1 logic threshold for search 1 is set to 2.0 Volts.

SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH<x>

This command sets or queries the math waveform threshold level for a pattern or state trigger search to determine where to place a mark. <x> is the search number or channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH<x>?
```

Arguments <NR3> specifies the threshold level in volts.

Examples SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:MATH2 2.0 sets the Math 2 waveform logic trigger threshold for search 1 to 2.0 Volts.

SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:MATH1 1.4000, indicating the Math 1 waveform logic threshold for search 1 is set to 1.4 Volts.

SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x>

This command sets or queries the reference waveform threshold level for a pattern or state trigger search to determine where to place a mark. <x> is the search number or the reference waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x>?
```

Arguments <NR3> specifies the threshold level in volts.

Examples SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:REF2 2.0 sets the Reference 2 waveform logic trigger threshold for search 1 to 2.0 Volts.

SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:LOGIC:THRESHOLD:REF1 1.4000, indicating the Reference 1 waveform logic trigger threshold for search 1 is set to 1.4 Volts.

SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn

This command sets or queries the pattern trigger pattern test that determines where to place a mark. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn
{TRUe|FALSe|LESSThan|MOREThan}
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn?
```

Related Commands SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:LESSLimit, SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:MORELimit

Arguments

TRUe specifies a TRUE logic level.

FALSe specifies a FALSE logic level.

LESSThan specifies the pattern must be TRUE less than the time limit.

MOREThan specifies the pattern must be TRUE more than the time limit.

Examples

SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN TRUE sets the pattern trigger to trigger when the pattern is TRUE.

SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN? might return :SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN TRUE, indicating search 1 will place a mark when the pattern trigger pattern is TRUE.

SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:LESSLimit

This command sets or queries the time the pattern trigger lesslimit test uses to determines where to place a mark. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:LESSLimit <NR3>
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:LESSLimit?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn, SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:MORELimit

Arguments

<NR3> is the time in seconds.

Examples

SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN:LESSLIMIT 1.0e-9 sets the less than time limit to 1 ns.

SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN:LESSLIMIT? might return :SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN:LESSLIMIT 1.0000E-9, indicating the lesslimit time is set to 1 ns.

SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:MORELimit

This command sets or queries the time the pattern trigger morelimit test uses to determines where to place a mark. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:MORELimit <NR3>
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:MORELimit?
```

Related Commands

SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn, SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:LESSLimit

Arguments

<NR3> is the time in seconds.

Examples

SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN:MORELIMIT 2.0e-9 sets the more limit time to 2 ns.

SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN:MORELIMIT? might return :SEARCH:SEARCH1:TRIGGER:A:PATTERN:WHEN:MORELIMIT 500.0000E-12, indicating the more limit time is set to 500 ps.

SEARCH:SEARCH<x>:TRIGger:A:PULse:SOUrce

This command sets or queries the source waveform for a pulse trigger search to determine where to place a mark. <x> is the search number.

NOTE. *This is the source for Glitch, Width, Runt, Timeout or Window searches.*

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:PULse:SOUrce
{CH<x>|MATH<x>|REF<x>}
SEARCH:SEARCH<x>:TRIGger:A:PULse:SOUrce?
```

Arguments

CH<x> specifies one input channel as the edge source, where <x> = 1, 2, 3 or 4.

MATH<x> specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

REF<x> specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

Examples

SEARCH:SEARCH1:TRIGGER:A:PULSE:SOURCE REF1 sets the pulse trigger source for search1 to the Reference 1 waveform.

SEARCH:SEARCH1:TRIGGER:A:PULSE:SOURCE? might return :SEARCH:SEARCH1:TRIGGER:A:PULSE:SOURCE MATH1, indicating the pulse trigger source for search 1 is set to the Math 1 waveform.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:CH<x>

This command sets or queries the polarity setting for a runt trigger search to determine where to place a mark. <x> is the search number or channel number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:CH<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:CH<x>?
```

Arguments

EITher places a mark on a runt of either polarity.

NEGative places a mark when the falling edge crosses the high threshold and the rising edge recrosses the high threshold without either edge ever crossing the low threshold.

POSitive places a mark when the rising edge crosses the low threshold and the falling edge recrosses the low threshold without either edge ever crossing the high threshold.

Examples

SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:CH2 NEGATIVE sets the Channel 2 runt polarity for search 1 to negative.

SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:CH1 POSITIVE, indicating the Channel 1 runt polarity for search 1 is set to positive.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:MATH<x>

This command sets or queries the polarity setting for a runt trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:MATH<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:MATH<x>?
```

Arguments

EITher places a mark on a runt of either polarity.

NEGative places a mark when the falling edge crosses the high threshold and the rising edge recrosses the high threshold without either edge ever crossing the low threshold.

POSitive places a mark when the rising edge crosses the low threshold and the falling edge recrosses the low threshold without either edge ever crossing the high threshold.

Examples

SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:MATH2 NEGATIVE sets the Math 2 waveform runt polarity for search 1 to negative.

SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:MATH1 POSITIVE, indicating the Math 1 waveform runt polarity for search 1 is set to positive.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:REF<x>

This command sets or queries the polarity setting for a runt trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:REF<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:REF<x>?
```

Arguments

`EITher` places a mark on a runt of either polarity.

`NEGative` places a mark when the falling edge crosses the high threshold and the rising edge recrosses the high threshold without either edge ever crossing the low threshold.

`POSitive` places a mark when the rising edge crosses the low threshold and the falling edge recrosses the low threshold without either edge ever crossing the high threshold.

Examples

`SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:REF2 NEGATIVE` sets the Reference 1 waveform runt polarity for search 1 to negative.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:POLARITY:REF1 POSITIVE`, indicating the Reference 1 waveform runt polarity for search 1 is set to positive.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:CH<x>

This command sets or queries the channel threshold HIGH level for a runt trigger search to determine where to place a mark. <x> is the search number or the channel number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:CH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:CH<x>?
```

Arguments

<NR3> specifies the HIGH threshold in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:CH2 1.3` sets the runt threshold high level to 1.3 Volts.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:CH1 2.0000`, indicating the Channel 1 runt trigger high threshold is set to 2.0 Volts.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:MATH<x>

This command sets or returns the math waveform threshold HIGH level for a runt trigger search to determine where to place a mark. <x> is the search number or the math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:MATH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:MATH<x>?
```

Arguments <NR3> specifies the HIGH threshold level.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:MATH2 1.3` sets the runt trigger HIGH threshold to 1.3 Volts.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:MATH1 2.0000`, indicating that the Math 1 waveform runt trigger HIGH threshold is set to 2.0 Volts.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:REF<x>

This command sets or returns the reference waveform threshold HIGH level for a runt trigger search to determine where to place a mark. <x> is the search number or the reference waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:REF<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:REF<x>?
```

Arguments <NR3> specifies the HIGH threshold in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:REF2 1.3` sets the runt trigger HIGH threshold to 1.3 Volts.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:HIGH:REF1 2.0000`, indicating that the Reference 1 waveform runt trigger HIGH threshold is set to 2.0 Volts.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:CH<x>

This command sets or queries the channel waveform threshold LOW level for a runt trigger search to determine where to place a mark. <x> is the search number or the channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:CH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:CH<x>?
```

Arguments <NR3> specifies the LOW threshold in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:CH2 0.10` sets the Channel 2 LOW runt trigger threshold for search 1 to 100 mV.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:CH1 800.0000E-3`, indicating the Channel 1 runt trigger LOW threshold for search 1 is set to 800 mV.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:MATH<x>

This command sets or queries the math waveform threshold LOW level for a runt trigger search to determine where to place a mark. <x> is the search number or the math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:MATH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:MATH<x>?
```

Arguments <NR3> specifies the LOW threshold level.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:MATH2 0.10` sets the Math 2 waveform runt trigger LOW threshold for search 1 to 100 mV.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:MATH1 800.0000E-3`, indicating the Math 1 waveform runt trigger LOW threshold for search 1 is set to 800 mV.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:REF<x>

This command sets or returns the reference waveform threshold LOW level for a runt trigger search to determine where to place a mark. <x> is the search number or the reference waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:REF<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:LOW:REF<x>?
```

Arguments <NR3> specifies the LOW threshold in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:REF2 0.10` sets the Reference 2 waveform runt trigger LOW threshold for search 1 to 100 mV.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:THRESHOLD:LOW:REF1 800.0000E-3`, indicating that the Reference 1 waveform runt trigger LOW threshold for search 1 is 800 mV.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn

This command 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 and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn {OCCurs|WIDERthan}
SEARCH:SEARCH<x>:TRIGger:A:RUNT:WHEn?
```

Arguments `OCCurs` argument specifies a trigger event if a runt of any detectable width occurs.

`WIDERthan` argument sets the oscilloscope to trigger if a runt pulse is detected with width wider than the time set by the SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth command.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:WHEN WIDERTHAN` sets the oscilloscope to trigger when a runt pulse is detected with width wider than the time set by the SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth command.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:WHEN?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:WHEN OCCURS`, indicating that a mark is placed if a runt trigger event occurs.

SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth

This command sets or queries the width setting for a runt trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth <NR3>
SEARCH:SEARCH<x>:TRIGger:A:RUNT:WIDth?
```

Arguments `<NR3>` specifies the minimum width in seconds.

Examples `SEARCH:SEARCH1:TRIGGER:A:RUNT:WIDTH 400E-12` sets the runt trigger minimum width for search 1 to 0.4 nanoseconds.

`SEARCH:SEARCH1:TRIGGER:A:RUNT:WIDTH?` might return `:SEARCH:SEARCH1:TRIGGER:A:RUNT:WIDTH 500.0000E-12`, indicating that the runt trigger minimum width for search 1 is set to 0.5 nanoseconds.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:EDGE

This command sets or queries the clock slope setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:EDGE {FALL|RISe}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:EDGE?
```

Arguments

FALL specifies the polarity as the clock falling edge.

RISe specifies the polarity as the clock rising edge.

Examples

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:EDGE FALL sets the setup/hold trigger clock slope polarity for search 1 to falling edge.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:EDGE? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:EDGE RISE, indicating that the setup/hold trigger clock slope polarity for search 1 is set to rising edge.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:SOUrce

This command sets or queries the clock source setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:SOUrce
{CH<x>|MATH<x>|REF<x>}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:SOUrce
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:SOUrce?
```

Arguments

CH<x> specifies an input channel as the edge source, where <x> = 1, 2, 3 or 4.

MATH<x> specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

REF<x> specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

Examples

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:SOURCE MATH1 sets the setup/hold trigger clock source setting for search 1 to MATH1.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:SOURCE? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:SOURCE CH1, indicating that the setup/hold trigger clock source setting for search 1 is set to CH1.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:CH<x>

This command sets or queries the clock threshold setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number or channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:CH<x>
{NR3|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:CH<x>?
```

Arguments <NR3> is the clock 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 SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:CH2 ECL sets the Channel 2 setup/hold trigger clock threshold setting for search 1 to ECL.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:CH1 −1.3000, indicating that the Channel 1 setup/hold trigger clock threshold setting for search 1 is set to -1.3 volts.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:MATH<x>

This command sets or queries the clock threshold setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:MATH<x>
{NR3|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:MATH<x>?
```

Arguments <NR3> is the clock 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 SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:MATH2 ECL sets the clock threshold setting of the Math 2 waveform for a setup/hold trigger search to ECL.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:MATH1

1.4000, indicating that the clock threshold setting of the Math 1 waveform for a setup/hold trigger search is set to 1.4 volts.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:REF<x>

This command sets or queries the clock threshold setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:REF<x>
{NR3|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:CLOCk:THReshold:REF<x>?
```

Arguments <NR3> is the clock 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 SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:REF2 ECL sets the clock threshold setting of the Reference 2 waveform for a setup/hold trigger search to ECL.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:CLOCK:THRESHOLD:REF1 1.0000, indicating that the clock threshold setting of the Reference 1 waveform for a setup/hold trigger search is set to 1 volt.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:SOUrce

This command sets or queries the data source setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:SOUrce
{CH<x>|MATH<x>|REF<x>}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:SOUrce?
```

Arguments

CH<x> specifies an input channel as the search source, where <x> = 1, 2, 3 or 4.

MATH<x> specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

REF<x> specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

Examples

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:SOURCE MATH1 sets the setup/hold trigger data source setting for search 1 to the Math 1 waveform.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:SOURCE? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:SOURCE CH1, indicating that the setup/hold trigger data source setting for search 1 is set to the Channel 1.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:CH<x>

This command sets or queries the data threshold setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number or channel number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:CH<x>
{NR3|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:CH<x>?
```

Arguments

<NR3> is the clock 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

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:CH2 ECL sets the Channel 2 setup/hold trigger data threshold setting for search 1 to -1.3 volts.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:CH1 1.4000, indicating that the Channel 1 setup/hold trigger data threshold setting for search 1 is set to 1.4 V.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:MATH<x>

This command sets or queries the data threshold setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:MATH<x>
{NR3|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:MATH<x>?
```

Arguments <NR3> is the clock 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 SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:MATH1 ECL sets the Math 2 waveform setup/hold trigger data threshold setting for search 1 to -1.3 volts.

SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:MATH1 1.0000, indicating that the Math 1 waveform setup/hold trigger data threshold setting for search 1 is set to 1.0 volt.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:REF<x>

This command sets or queries the data threshold setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:REF<x>
{NR3|ECL|TTL}
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:THReshold:REF<x>?
```

Arguments <NR3> is the clock 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 `SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:REF2 ECL` sets the Reference 2 waveform setup/hold trigger data threshold setting for search 1 to -1.3 volts.

`SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:SETHOLD:DATA:THRESHOLD:REF1 1.4000`, indicating that the Reference 1 waveform setup/hold trigger data threshold setting for search 1 is set to 1.4 volts.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:HOLDTime

This command sets or queries the hold time setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

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.

Examples `SEARCH:SEARCH1:TRIGGER:A:SETHOLD:HOLDTIME 400E-12` sets the setup/hold trigger hold time setting for search 1 to 400 ps.

`SEARCH:SEARCH1:TRIGGER:A:SETHOLD:HOLDTIME?` might return `:SEARCH:SEARCH1:TRIGGER:A:SETHOLD:HOLDTIME 500.0000E-12`, indicating that the setup/hold trigger hold time setting for search 1 is set to 0.5 ns.

SEARCH:SEARCH<x>:TRIGger:A:SETHold:SETTime

This command sets or queries the setup time setting for a setup/hold trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

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.

Examples `SEARCH:SEARCH1:TRIGGER:A:SETHOLD:SETTIME 2E-9` sets the setup/hold trigger setup time setting for search 1 to 2 ns.

`SEARCH:SEARCH1:TRIGGER:A:SETHOLD:SETTIME?` might return `:SEARCH:SEARCH1:TRIGGER:A:SETHOLD:SETTIME 1.0000E-9`, indicating that the setup/hold trigger setup time setting for search 1 is set to 1.0 ns.

SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:EDGE

This command sets or queries the clock slope setting for a state trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:EDGE {FALL|RISe}
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:EDGE?
```

Arguments `FALL` specifies polarity as the clock falling edge.

`RISe` specifies polarity as the clock rising edge.

Examples `SEARCH:SEARCH1:TRIGGER:A:STATE:CLOCK:EDGE RISE` sets the state trigger clock slope polarity setting for search 1 to RISE.

`SEARCH:SEARCH1:TRIGGER:A:STATE:CLOCK:EDGE?` might return `:SEARCH:SEARCH1:TRIGGER:A:STATE:CLOCk:EDGE FALL`, indicating that the state trigger clock slope polarity setting for search 1 is set to FALL.

SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:SOUrce

This command sets or queries the clock source of the twelve waveforms for a state trigger search to determine where to place a mark. <x> is the search number. The hardware trigger is always on the channel 4.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:SOUrce
{CH<x>|MATH<x>|REF<x>}
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:SOUrce?
```

Arguments

CH<x> specifies an input channel as the search source, where <x> = 1, 2, 3 or 4.

MATH<x> specifies the math waveform as the search source, where <x> = 1, 2, 3 or 4.

REF<x> specifies the reference waveform as the search source, where <x> = 1, 2, 3 or 4.

Examples

SEARCH:SEARCH1:TRIGGER:A:STATE:CLOCK:SOURCE CH2 sets the state trigger clock source for search 1 to Channel 2.

SEARCH:SEARCH1:TRIGGER:A:STATE:CLOCK:SOURCE? might return :SEARCH:SEARCH1:TRIGGER:A:STATE:CLOCK:SOURCE CH1, indicating that the state trigger clock source for search 1 is set to Channel 1.

SEARCH:SEARCH<x>:TRIGger:A:STATE:WHEn

This command sets or queries the state trigger search to determine where to place a mark. <x> is the search number.

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:STATE:WHEn {TRUe|FALSe}
SEARCH:SEARCH<x>:TRIGger:A:STATE:WHEn?

Arguments

TRUe specifies a TRUE logic level.

FALSe specifies a FALSE logic level.

Examples

SEARCH:SEARCH1:TRIGGER:A:STATE:WHEN FALSE sets the state trigger logic level for search 1 to FALSE.

SEARCH:SEARCH1:TRIGGER:A:STATE:WHEN? might return :SEARCH:SEARCH1:TRIGGER:A:STATE:WHEN TRUE, indicating that the state trigger logic level for search 1 is set to TRUE.

SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:CH<x>

This command sets or queries the polarity for a timeout trigger search to determine where to place a mark. <x> is the search number or channel number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:CH<x>
{STAYSHigh|STAYSLow|EITher}
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:CH<x>?
```

Arguments

`STAYSHigh` specifies the polarity stays HIGH.

`STAYSLow` specifies the polarity stays LOW.

`EITher` specifies the polarity either stays HIGH or stays LOW.

Examples

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:CH2 STAYSLOW` sets the Channel 2 timeout trigger polarity for search 1 to STAYSLOW.

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:CH1 STAYSHIGH`, indicating that the Channel 2 timeout trigger polarity for search 1 is set to STAYSHIGH.

SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:MATH<x>

This command sets or queries the polarity for a timeout trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:MATH<x>
{STAYSHigh|STAYSLow|EITher}
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:MATH<x>?
```

Arguments

`STAYSHigh` specifies the polarity stays HIGH.

`STAYSLow` specifies the polarity stays LOW.

`EITher` specifies the polarity either stays HIGH or stays LOW.

Examples

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:MATH2 STAYSLOW` sets the Math 2 waveform timeout trigger polarity for search 1 to STAYSLOW.

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:MATH1 STAYSHIGH`, indicating that the Math 1 waveform timeout trigger polarity for search 1 is set to STAYSHIGH.

SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:REF<x>

This command sets or queries the polarity for a timeout trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:REF<x>
{STAYSHigh|STAYSLow|EITher}
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:POLarity:REF<x>?
```

Arguments

`STAYSHigh` specifies the polarity stays HIGH.

`STAYSLow` specifies the polarity stays LOW.

`EITher` specifies the polarity either stays HIGH or stays LOW.

Examples

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:REF2 STAYSLOW` sets the Reference 2 waveform timeout trigger polarity for search 1 to STAYSLOW.

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:POLARITY:REF1 STAYSHIGH`, indicating that the Reference 2 waveform timeout trigger polarity for search 1 is set to STAYSHIGH.

SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:TIMe

This command sets or queries the time setting for a timeout trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:TIMe <NR3>
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:TIMe?
```

Arguments

`<NR3>` is the time in seconds.

Examples

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:TIME 400E-9` sets the timeout trigger time setting for search 1 to 400 ns.

`SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:TIME?` might return `:SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:TIME 500.0000E-12`, indicating that the timeout trigger time setting for search 1 is set to 500 ns.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:DELTATime

This command sets or queries the transition time setting for a transition trigger search to determine where to place a mark.<x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:DELTATime <NR3>
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:DELTATime?
```

Arguments <NR3> specifies the transition time in seconds.

Examples SEARCH:SEARCH1:TRIGGER:A:TIMEOUT:TIME 400E-9 sets the transition trigger time setting for search 1 to 400 ns.

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:DELTATIME? might return :SEARCH:SEARCH1:TRIGGER:A:TRANSITION:DELTATIME 500.0000E-12, indicating that the transition trigger time setting for search 1 is set to 500 ps.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:CH<x>

This command sets or queries the polarity setting for a transition trigger search to determine where to place a mark. <x> is the search number or channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:CH<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:CH<x>?
```

Arguments EITher specifies either positive or negative polarity.

NEGative specifies that a pulse edge must traverse from the upper (most positive) to lower (most negative) level for transition triggering to occur.

POSitive specifies that a pulse edge must traverse from the lower (most negative) to higher (most positive) level for transition triggering to occur.

Examples SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:CH12 NEGATIVE sets the Channel 2 transition trigger polarity setting for search 1 to NEGATIVE.

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:CH1 POSITIVE,

indicating that the Channel 1 transition trigger polarity setting for search 1 is set to POSITIVE.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:MATH<x>

This command sets or queries the polarity setting for a transition trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:MATH<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:MATH<x>?
```

Arguments `EITher` specifies either positive or negative polarity.

`NEGative` specifies that a pulse edge must traverse from the upper (most positive) to lower (most negative) level for transition triggering to occur.

`POSitive` specifies that a pulse edge must traverse from the lower (most negative) to higher (most positive) level for transition triggering to occur.

Examples `SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:MATH2 NEGATIVE` sets the Math 2 waveform transition trigger polarity setting for search 1 to NEGATIVE.

`SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:MATH1 POSITIVE`, indicating that the Math 1 waveform transition trigger polarity setting for search 1 is set to POSITIVE.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:REF<x>

This command sets or queries the polarity setting for a transition trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:REF<x>
{EITher|NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:POLarity:REF<x>?
```

Arguments

EITher specifies either positive or negative polarity.

NEGative specifies that a pulse edge must traverse from the upper (most positive) to lower (most negative) level for transition triggering to occur.

POSitive specifies that a pulse edge must traverse from the lower (most negative) to higher (most positive) level for transition triggering to occur.

Examples

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:REF2 NEGATIVE sets the Reference 2 waveform transition trigger polarity setting for search 1 to NEGATIVE.

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:TRANSITION:POLARITY:REF1 POSITIVE, indicating that the Reference 2 waveform transition trigger polarity setting for search 1 is set to POSITIVE.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:CH<x>

This command sets or queries the threshold high voltage level setting for a transition trigger search to determine where to place a mark. <x> is the search number or channel number.

Group

Search and Mark

Syntax

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:CH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:CH<x>?

Arguments

<NR3> specifies the HIGH threshold in volts.

Examples

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:CH2 1.5 sets the Channel 2 transition trigger HIGH threshold setting for search 1 to 1.5 volts.

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:CH1 2.0000, indicating that the Channel 1 transition trigger HIGH threshold setting for search 1 is set to 2 volts.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:MATH<x>

This command sets or queries the threshold high voltage level setting for a transition trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:MATH<x>
<NR3>
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:
MATH<x>?
```

Arguments <NR3> specifies the HIGH threshold in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:MATH2 1.5` sets the Math 2 waveform transition trigger HIGH threshold setting for search 1 to 1.5 volts.

`SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:MATH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:MATH1 2.0000`, indicating that the Math 1 waveform transition trigger HIGH threshold setting for search 1 is set to 2 volts.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:REF<x>

This command sets or queries the threshold high voltage level setting for a transition trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:REF<x>
<NR3>
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:HIGH:REF<x>?
```

Arguments <NR3> specifies the HIGH threshold in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:REF2 1.5` sets the Reference 2 waveform transition trigger HIGH threshold setting for search 1 to 1.5 volts.

`SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:HIGH:REF1 2.0000`, indicating that the Reference 1 waveform transition trigger HIGH threshold setting for search 1 is set to 2 volts.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:CH<x>

This command sets or queries the threshold low voltage level setting for a transition trigger search to determine where to place a mark. <x> is the search number or channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:CH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:CH<x>?
```

Arguments <NR3> specifies the LOW threshold in volts.

Examples `SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:CH2 0.5` sets the Channel 2 transition trigger LOW threshold setting for search 1 to 0.5 volts.

`SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:CH1 800.0000E-3`, indicating that the Channel 1 transition trigger LOW threshold setting for search 1 is set to 0.8 volts.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:MATH<x>

This command sets or queries the threshold low voltage level setting for a transition trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:MATH<x> <NR3>
```

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:MATH<x>?

Arguments <NR3> specifies the LOW threshold in volts.

Examples SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:MATH2 0.5 sets the Math 2 waveform transition trigger LOW threshold setting for search 1 to 0.5 volts.

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:MATH1 800.0000E-3, indicating that the Math 1 waveform transition trigger LOW threshold setting for search 1 is set to 0.8 volts.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:REF<x>

This command sets or queries the threshold low voltage level setting for a transition trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:REF<x>
<NR3>
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:THReshold:LOW:REF<x>?
```

Arguments <NR3> specifies the LOW threshold in volts.

Examples SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:REF2 0.5 sets the Reference 2 waveform transition trigger LOW threshold setting for search 1 to 0.5 volts.

SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:TRANSITION:THRESHOLD:LOW:REF1 800.0000E-3, indicating that the Reference 1 waveform transition trigger LOW threshold setting for search 1 is set to 0.8 volts.

SEARCH:SEARCH<x>:TRIGger:A:TRANsition:WHEn

This command sets or queries the condition setting for a transition trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:WHEn
{FASTERthan|SLOWERthan}
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:WHEn?
```

Arguments `FASTERthan` sets the trigger to occur when the transitioning signal is faster than the set volts/second rate.

`SLOWERthan` sets the trigger to occur when the transitioning signal is slower than the set volts/second rate.

Examples `SEARCH:SEARCH1:TRIGGER:A:TRANSITION:WHEN SLOWERTHAN` sets the transition trigger condition setting for search 1 to SLOWERTHAN.

`SEARCH:SEARCH1:TRIGGER:A:TRANSITION:WHEN?` might return `:SEARCH:SEARCH1:TRIGGER:A:TRANSITION:WHEN FASTERTHAN`, indicating that the transition trigger condition setting for search 1 is set to FASTERTHAN.

SEARCH:SEARCH<x>:TRIGger:A:TYPe

This command sets or queries the trigger type setting for a search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:TYPe
{EDGE|RUNT|TRANsition|PATtern|GLItch|SETHold|UNDEFINED|
WIDth|TIMEOut|WINdow|STATE|DDRRead|DDRWrite|DDRREADWrite}
SEARCH:SEARCH<x>:TRIGger:A:TYPe?
```

Arguments `EDGE` triggers when the source input signal amplitude crosses the specified level in the direction given by the slope.

NOTE. *Some trigger types are optional.*

`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 SEARCH:SEARCH<x>:TRIGger:A:RUNt:HIGH and SEARCH:SEARCH<x>:TRIGger:A:RUNt:LOW THRESHOLD commands.

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.

PATtern trigger on a functional pattern combination of one to four input signal sources.

GLItch triggers at the trailing edge of the glitch pulse where the signal crosses the trigger level.

SETHold triggers on the data input signal transitions relative to the clock input signal transitions within the setup and hold time violation zone.

UNDEFINED is the default. To perform a search, select a different argument.

WIDth triggers on input signal source pulses that are inside or outside of the given time range.

TIMEOut triggers on an input signal source that stays above, stays below, or stays either above or beow the trigger level for a given time.

WINdow triggers on an input signal source that enters or exits the horizontal band defined by the two trigger levels.

STATE triggers on a functional pattern combination of one to three data sources at the time of the clock transition.

DDRRead triggers at the start and end of the read bus.

DDRWrite triggers at the start and end of the write bus.

DDRREADWrite triggers at the start and end of both the read and write bus.

Examples

SEARCH:SEARCH1:TRIGGER:A:TYPE RUNT sets the trigger type setting for search 1 to RUNT.

SEARCH:SEARCH1:TRIGGER:A:TYPE? might return :SEARCH:SEARCH1:TRIGGER:A:TYPE EDGE, indicating that the trigger type setting for search 1 is set to EDGE.

SEARCH:SEARCH<x>:TRIGger:A:WIDth:HIGHLimit

This command sets or queries the upper time limit setting for a width trigger search to determine where to place a mark. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:WIDth:HIGHLimit <NR3>
SEARCH:SEARCH<x>:TRIGger:A:WIDth:HIGHLimit?
```

Arguments <NR3> specifies the upper time limit in seconds.

Examples SEARCH:SEARCH1:TRIGGER:A:WIDTH:HIGHLIMIT 400E-12 sets the width trigger upper time limit setting for search 1 to 0.4 ns.

SEARCH:SEARCH1:TRIGGER:A:WIDTH:HIGHLIMIT? might return :SEARCH:SEARCH1:TRIGGER:A:WIDTH:HIGHLIMIT 500.0000E-12, indicating that the width trigger upper time limit setting for search 1 is set to 0.5 ns.

SEARCH:SEARCH<x>:TRIGger:A:WIDth:LOWLimit

This command sets or queries the lower time limit setting for a width trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WIDth:LOWLimit <NR3>
SEARCH:SEARCH<x>:TRIGger:A:WIDth:LOWLimit?
```

Arguments <NR3> specifies the lower time limit in seconds.

Examples SEARCH:SEARCH1:TRIGGER:A:WIDTH:LOWLIMIT 400E-12 sets the width trigger lower time limit setting for search 1 to 0.4 ns.

SEARCH:SEARCH1:TRIGGER:A:WIDTH:LOWLIMIT? might return :SEARCH:SEARCH1:TRIGGER:A:WIDTH:LOWLIMIT 500.0000E-12, indicating that the width trigger lower time limit setting for search 1 is set to 0.5 ns.

SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:CH<x>

This command sets or queries the polarity for a width trigger search to determine where to place a mark. <x> is the search number or channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:CH<x>
{NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:CH<x>?
```

Arguments NEGAtive specifies a negative polarity.

POSITIVe specifies a positive polarity.

Examples SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:CH2 NEGATIVE sets the Channel 2 width trigger polarity setting for search 1 to NEGATIVE.

SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:CH1 POSITIVE, indicating that the Channel 1 width trigger polarity setting for search 1 is set to POSITIVE.

SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:MATH<x>

This command sets or queries the polarity for a width trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:MATH<x>
{NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:MATH<x>?
```

Arguments NEGAtive specifies a negative polarity.

POSITIVe specifies a positive polarity.

Examples SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:MATH2 NEGATIVE sets the Math 2 waveform width trigger polarity setting for search 1 to NEGATIVE.

SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:MATH1 POSITIVE, indicating that the Math 1 waveform width trigger polarity setting for search 1 is set to POSITIVE.

SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:REF<x>

This command sets or queries the polarity for a width trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:REF<x>
{NEGAtive|POSITIVe}
SEARCH:SEARCH<x>:TRIGger:A:WIDth:POLarity:REF<x>?
```

Arguments

`NEGAtive` specifies a negative polarity.

`POSITIVe` specifies a positive polarity.

Examples

`SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:REF2 NEGATIVE` sets the Reference 2 waveform width trigger polarity setting for search 1 to NEGATIVE.

`SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:WIDTH:POLARITY:REF1 POSITIVE`, indicating that the Reference 1 waveform width trigger polarity setting for search 1 is set to POSITIVE.

SEARCH:SEARCH<x>:TRIGger:A:WIDth:WHEn

This command sets or queries the width trigger search to determine where to place a mark. <x> is the search number.

Group

Search and Mark

Syntax

```
SEARCH:SEARCH<x>:TRIGger:A:WIDth:WHEn {OUTside|WIThin}
SEARCH:SEARCH<x>:TRIGger:A:WIDth:WHEn?
```

Arguments

`OUTside` specifies the outside limits.

`WIThin` specifies the inside limits.

Examples

`SEARCH:SEARCH1:TRIGGER:A:WIDTH:WHEN OUTSIDE` sets the width trigger for search 1 to outside limits.

`SEARCH:SEARCH1:TRIGGER:A:WIDTH:WHEN?` might return `:SEARCH:SEARCH1:TRIGGER:A:WIDTH:WHEN WITHIN`, indicating that the width trigger for search 1 is set to inside limits.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:EVENT

This command sets or queries the window trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:EVENT
{OUTSIDEGreater|INSIDEGreater|ENTERSWindow|EXITSWindow}
SEARCH:SEARCH<x>:TRIGger:A:WINdow:EVENT?
```

Arguments

`OUTSIDEGreater` specifies a trigger event when the signal leaves the window defined by the threshold levels for the time specified by Width.

`INSIDEGreater` specifies a trigger event when the signal enters the window defined by the threshold levels for the time specified by Width.

`ENTERSWindow` specifies a trigger event when the signal enters the window defined by the threshold levels.

`EXITSWindow` specifies a trigger event when the signal leaves the window defined by the threshold levels.

Examples

`SEARCH:SEARCH1:TRIGGER:A:WIDTH:EVENT OUTSIDEGREATER` sets the window trigger for search 1 specifies a trigger when the signal leaves the window defined by the threshold levels for the time specified by width.

`SEARCH:SEARCH1:TRIGGER:A:WIDTH:EVENT?` might return `:SEARCH:SEARCH1:TRIGGER:A:WIDTH:EVENT ENTERSWINDOW`, indicating that the window trigger for search 1 specifies a trigger when the signal enters the window defined by the threshold levels.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:CH<x>

This command sets or queries the channel threshold HIGH level for a window trigger search to determine where to place a mark. <x> is the search number or channel number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:CH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:CH<x>?
```

Arguments

`<NR3>` specifies the HIGH threshold level.

Examples

`SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:CH2 1.5000` sets the Channel 2 window trigger high threshold level for search 1 to 1.5 volts.

SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:CH1? might return :SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:CH1 2.0000, indicating that the Channel 2 window trigger high threshold level for search 1 is set to 2.0 V.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:MATH<x>

This command sets or queries the math waveform threshold HIGH level for a window trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:MATH<x>
<NR3>
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:MATH<x>?
```

Arguments <NR3> specifies the HIGH threshold level.

Examples SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:MATH2 1.5000 sets the Math 2 waveform window trigger high threshold level for search 1 to 1.5 volts.

SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:MATH1 2.0000, indicating that the Math 1 waveform window trigger high threshold level for search 1 is set to 2.0 volts.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:REF<x>

This command sets or queries the reference waveform threshold HIGH level for a window trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:REF<x>
<NR3>
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:HIGH:REF<x>?
```

Arguments <NR3> specifies the HIGH threshold level.

Examples `SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:REF2 1.5000` sets the Reference 2 waveform window trigger high threshold level for search 1 to 1.5 volts.

`SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:REF1?` might return `:SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:HIGH:REF1 2.0000`, indicating that the Reference 1 waveform window trigger high threshold level for search 1 is set to 2.0 volts.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:CH<x>

This command sets or queries the channel waveform threshold LOW level for a window trigger search to determine where to place a mark. <x> is the search number or channel waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:CH<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:CH<x>?
```

Arguments `<NR3>` specifies the LOW threshold level.

Examples `SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:CH2 0.5` sets the Channel 2 window trigger low threshold level for search 1 to 0.5 volts.

`SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:CH1?` might return `:SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:CH1 800.0000E-3`, indicating that the Channel 1 window trigger high threshold level for search 1 is set to 0.8 volts.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:MATH<x>

This command sets or queries the math waveform threshold LOW level for a window trigger search to determine where to place a mark. <x> is the search number or math waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:MATH<x>
<NR3>
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:MATH<x>?
```

Arguments <NR3> specifies the LOW threshold level.

Examples SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:MATH2 0.5 sets the Math 2 waveform window trigger low threshold level for search 1 to 0.5 volts.

SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:MATH1? might return :SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:MATH1 800.0000E-3, indicating that the Math 1 waveform window trigger low threshold level for search 1 is set to 0.8 volts.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:REF<x>

This command sets or queries the reference waveform threshold LOW level for a window trigger search to determine where to place a mark. <x> is the search number or reference waveform number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:REF<x> <NR3>
SEARCH:SEARCH<x>:TRIGger:A:WINdow:THReshold:LOW:REF<x>?
```

Arguments <NR3> specifies the LOW threshold level.

Examples SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:REF2 0.5 sets the Reference 2 waveform window trigger low threshold level for search 1 to 0.5 volts.

SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:REF1? might return :SEARCH:SEARCH1:TRIGGER:A:WINDOW:THRESHOLD:LOW:REF1 2.0000, indicating that the Reference 1 waveform window trigger low threshold level for search 1 is set to 0.8 volts.

SEARCH:SEARCH<x>:TRIGger:A:WINdow:WIDth

This command sets or queries the width setting for a window trigger search to determine where to place a mark. <x> is the search number.

Group Search and Mark

Syntax
```
SEARCH:SEARCH<x>:TRIGger:A:WINdow:WIDth <NR3>
SEARCH:SEARCH<x>:TRIGger:A:WINdow:WIDth?
```

Arguments <NR3> specifies the minimum width in seconds.

ECL specifies a preset high level of -1.3 V and TTL specifies a preset high level of +1.4V.

Examples SEARCH:SEARCH1:TRIGGER:A:WINDOW:WIDTH 400E-12 sets the window trigger width setting for search 1 to 0.4 ns.

SEARCH:SEARCH1:TRIGGER:A:WINDOW:WIDTH? might return :SEARCH:SEARCH1:TRIGGER:A:WINDOW:WIDTH 500.0000E-12, indicating that the window trigger width setting for search 1 is set to 0.5 ns.

SEARCH:STOP

This command sets or queries the stop of acquisitions when a search finds an event.

Group Search and Mark

Syntax
```
SEARCH:STOP {<NR1>|OFF|ON|1|0}
SEARCH:STOP?
```

Arguments <NR1> any numerical nonzero, stops the acquisition.

OFF continues the acquisition.

ON stops the acquisition.

1 stops the acquisition.

0 continues the acquisition.

Examples SEARCH:STOP 1 stops the acquisition.

SEARCH:STOP? might return :SEARCH:STOP 1, indicating that a search found an event and stops acquisition.

SELect? (Query Only)

This query-only command returns the selected waveform that is affected by the front panel controls and the display status (on or off) of all waveforms. This query command is equivalent to selecting Measurement Setup from the Measure menu and viewing the Source waveform setting.

Group Vertical

Syntax `SELect?`

Examples `SELECT?` might return the waveform and status information as `:SELECT:CH1 1;CH2 0;CH3 0;CH4 0;MATH1 0; MATH2 0;MATH3 0;MATH4 0;REF1 0;REF2 0; REF3 0;REF4 0;CONTROL CH1`.

SELect:B<x>

This command sets or queries the display state for the bus specified by x. The value of x can range from 1 through 16.

Group Bus

Syntax
```
SELect:B<x> {0|1|OFF|ON}
SELect:B<x>?
```

Related Commands SELect:CH<x>, SELect:D<x>, SELect:<wfm>

Arguments `0` turns off the display of the specified bus; any other value turns on the display of the specified bus.

`1` turns on the display of the specified bus.

`OFF` turns off the display of the indicated bus.

`ON` turns on the display of the indicated bus.

Examples `SELECT:B1 1` displays bus B1.

`SELECT:B1?` might return `:SELECT:B1 ON`, indicating bus B1 is displayed.

SELect:CH<x>

This command sets or queries the displayed state of the specified channel waveform. The x can be channel 1 through 4.

Group Vertical

Syntax
```
SELect:CH<x> {<NR1>|OFF|ON}
SELect:CH<x>?
```

Related Commands SELect:B<x>, SELect:D<x>, SELect:<wfm>

Arguments <NR1> = 0 turns off the display of the specified channel waveform; any other value turns on the display of the specified waveform.

OFF turns off the display of the indicated channel waveform.

ON displays the indicated channel waveform.

Examples SELECT:CH1 ON displays Channel 1.

SELECT:CH1? might return :SELECT:CH1 1, indicating Channel 1 is displayed.

SELect:CONTROl<wfm>

This command sets or queries the waveform that is selected as the implied recipient of channel-related commands that support legacy-style programs. The command form also performs the equivalent of a SELect:<wfm> ON command. This command is equivalent to selecting Measurement Setup from the Measure menu and either viewing or setting the Source waveform.

Group Vertical

Syntax
```
SELect:CONTROl {CH<x>|MATH<x>|REF<x>}
SELect:CONTROl?
```

Arguments CH<x> selects the specified channel waveform as the waveform that is affected by the front panel controls. The x variable can be expressed as an integer ranging from 1 through 4.

MATH<x> selects the specified math waveform as the waveform that is affected by the front panel controls. The x variable can be expressed as an integer ranging from 1 through 4.

REF<x> selects the specified reference waveform as the waveform that is affected by the front panel controls. The x variable can be expressed as an integer ranging from 1 through 4.

Examples SELECT:CONTROL CH2 resets acquisition displays on Channel 2, and causes the selected waveform to be the implied object of waveform commands.

SELECT:CONTROL? might return :SELECT:CONTROL MATH3, indicating that Math 3 is the implied object of waveform commands.

SELect:DALL (No Query Form)

This command sets the displayed state of all the digital inputs.

Group Digital

Syntax `SELect:DALL <NR1>`

Arguments `<NR1> = 0` turns off the display of all the digital inputs; any other value turns on the display of all the digital inputs.

`OFF` turns off the display of all the digital inputs.

`ON` displays all the digital inputs.

Examples `SELECT:DALL ON` sets the displayed state of all the digital inputs to on.

SELect:DIGTraces:COMbination (No Query Form)

This command turns on the digital channels that have binary digits as 1. The binary digits are obtained from the set decimal value.

Group Digital

Syntax `SELect:DIGTraces:COMbination <nr1>`

Arguments `nr1` is a decimal integer.

Examples `SELECT:DIGTRACES:COMBINATION 6` turns on the D1 and D2 digital channels.

SELect:DIGTraces:LISt

This command turns on the specified digital channels or returns the list of digital channels that are on.

Group Digital

Syntax
```
SELect:DIGTraces:LISt <Dx>
SELect:DIGTraces:LISt?
```

Arguments <Dx> are the specified digital channels, for example D3, D6, D8.

Examples `SELECT:DIGTRACES:LIST D0 D1` turns on the digital channel D0 and D1.

`SELECT:DIGTRACES:LIST?` might return `:SELECT:DIGTRACES:LIST D3, D4, D5`, indicating that digital channels D3, D4, and D5 are on. The other digital channels are off.

SELect:D<x>

This command sets or queries the display state for the digital input specified by x. The value of x can range from 0 through 15.

Group Digital

Syntax

```
SELect:D<x> {<NR1>|OFF|ON}
SELect:D<x>?
```

Related Commands SELect:B<x>, SELect:CH<x>, SELect:<wfm>

Arguments `<NR1> = 0` turns off the display of the specified digital input; any other value turns on the display of the specified bus.

`OFF` turns off the display of the indicated bus.

`ON` turns on the display of the indicated bus.

Examples `SELECT:D1 ON` displays digital input D1.

`SELECT:D1?` might return `:SELECT:D1 1`, indicating digital input D1 is displayed.

SELect:<wfm>

This command turns on the display of a specified waveform and also resets the acquisition. The query returns whether the channel is on or off but does not indicate whether it is the selected waveform. WFM can be a channel, math, or reference waveform.

Group Vertical

Syntax

```
SELect:<wfm> {ON|OFF|<NR1>}
SELect:<wfm>?
```

Related Commands

SELect:B<x>, SELect:CH<x>, SELect:D<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:REF1? might return :SELECT:REF1 1, indicating that Reference 1 waveform is being displayed.

SET? (Query Only)

This query-only command returns the commands that list the instrument settings, except for configuration information for the calibration values. You can use these commands to return the instrument 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 instrument 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.

Group

Miscellaneous

Syntax

```
SET?
```

Related Commands

HEADer, *LRN?, VERBose

Examples

SET? might return the following response: :ACQUIRE:STOPAFTER RUNSTOP;STATE 1;MODE SAMPLE;NUMENV 10;NUMAVG 16;REPET 1;:FASTACQ:STATE 0;:APPLICATION:GPKNOB1:ACTIVE 0;:APPLICATION:GPKNOB2:ACTIVE 0;:APPLICATION:WINDOW:HEIGHT 236;WIDTH 640;:APPLICATION:SCOPEAPP:STATE RUNNING;WINDOW FULLSCREEN;:APPLICATION:EXTAPP:STATE

```
NOTRUNNING;:AUXOUT:SOURCE ATRIGGER;EDGE FALLING;:CMDBATCH
1;:HEADER 1;:LOCK NONE;:ROSC:SOURCE INTERNAL;:VERBOSE
1;:ALIAS:STATE 0;:DISPLAY:CLOCK 1;COLOR:PALETTE
NORMAL;MATHCOLOR DEFAULT;REFCOLOR DEFAULT;:DISPLAY:FILTER
SINX;FORMAT YT;GRATICULE FULL;INTENSITY:WAVEFORM
75.0000;AUTOBRIGHT 1;:DISPLAY:PERSISTENCE
OFF;STYLE VECTORS;TRIGBAR SHORT;TRIGT 1;VARPERSIST
500.0000E-3;:HARDCOPY:FILENAME "untitled";PORT
FILE;:DIAG:LEVEL SUBSYS;:SAVE:WAVEFORM:FILEFORMAT
INTERNAL;:TRIGGER:A:MODE AUTO;TYPE
EDGE;LEVEL 0.0000;HOLDOFF:BY DEFAULT;TIME
1.5000E-6;:TRIGGER:A:EDGE:SOURCE CH1;COUPLING DC;SLOPE RISE;.
```

SETUp:NAMe

This command allows you to create (or query) a name for your saved setups. The default name for all user setups is “User.” The default name for factory setups is “Factory.”

This command is equivalent to selecting Save As from the File menu, pressing the Setup button, selecting the desired setup location, clicking the keyboard icon, and entering your setup name.

Group Save and Recall

Syntax

```
SETUp:NAMe <NR1>,<QString>
SETUp:NAMe?  <NR1>
```

Related Commands SAVe:SETUp

Arguments <NR1> specifies a location in which the setup label is stored. Location values range from 1 through 10.

<QString> is a string containing the setup label.

Examples SETUP:NAME1, "My Setup" changes the setup label for internal setup location 1 to “My Setup”.

SETUP:NAME? 2 might return :SETUP:NAME 2,"My Setup", indicating that “My Setup” is the setup label that you defined for internal setup location 2.

*SRE

The *SRE (Service Request Enable) command sets and queries the bits in the Service Request Enable Register. For more 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)

The *STB? (Read Status Byte) query returns the contents of the Status Byte Register (SBR) using the Master Summary Status (MSS) bit. For more 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, for the current user, both waveform and setup memories, overwriting any previously stored data.

TEKSecure deletes all four waveform reference memory slots on the hard drive, if they exist, and puts all setups in the factory-initialized state. External setups that are stored on the hard drive are not affected.

Group Miscellaneous

Syntax

```
TEKSecure
```

Examples `TEKSECURE` initializes both waveform and setup memories.

TEST

This command provides the ability to select and execute an item at any level of the test hierarchy (Test, Area or Subsystem). The query returns the last command sent. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu, choosing a test and then pressing Run.

NOTE. *We recommend turning off both DIAg:CONTROL:HALT and DIAg:CONTROL:LOOP before executing the TEST command.*

Group Diagnostics

Syntax

```
TEST <QString>
TEST?
```

Related Commands TEST:RESults?, TEST:RESults:VERBose?, TEST:STOP

Arguments `<QString>` sets the test ID, which ranges from 0 through 3 characters. If no test ID is specified, all available diagnostics are executed.

Examples `TEST "1"` executes all Acquisition tests (Subsystem:Acquisition).

`TEST "11"` executes all Acquisition Memory tests (Subsystem:Acquisition, Area:Memory).

TEST "113" executes a specific Acquisition Memory test (Subsystem:Acquisition, Area:Memory, Test:Address Lines).

TEST:RESults? (Query Only)

This query-only command returns an abbreviated status about the results of the last TEST execution. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then reviewing the Diagnostic Status.

Group Diagnostics

Syntax TEST:RESults?

Related Commands TEST, TEST:RESults:VERBose?

Examples TEST:RESULTS? might return :TEST:RESULTS "135", indicating the specific failure code of the first test failure.

TEST:RESULTS? might return :TEST:RESULTS "PASS", indicating that the previously executed test passed.

TEST:RESULTS? might return :TEST:RESULTS "*****", indicating that the test was not run.

TEST:RESults:VERBose? (Query Only)

This query-only command returns a more explanatory message about the results of the last TEST execution than the TEST:RESults? query. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then clicking the Error Log button.

Group Diagnostics

Syntax TEST:RESults:VERBose?

Related Commands TEST, TEST:RESults?

Examples TEST:RESULTS:VERBOSE? might return :TEST:RESULTS:VERBOSE DIAG ABORTED, ERROR 15 in A2D-Dmux 1 interconnects".

TEST:STOP (No Query Form)

This command (no query form) causes test (or diagnostics) execution to terminate at the end of the next low-level test. This command is equivalent to selecting Instrument Diagnostics from the Utilities menu and then clicking Abort.

Group Diagnostics

Syntax

```
TEST:STOP
```

Related Commands DIAg:STOP

Examples `TEST:STOP` terminates test (or diagnostics) execution at the end of the next low-level test.

TIME

This command sets or queries the time that the instrument displays. This command is equivalent to selecting Set Time & Date from the Utilities menu and then setting the fields in the Time group box.

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 01 to 60, and ss refers to a two-digit second number from 01 to 60.

Examples `TIME “14:00:00”` sets the time to exactly 2:00 p.m.

`TIME?` might return `:TIME "14:05:17`, indicating the current time is set to 2:05 p.m. and 17 seconds.

*TRG (No Query Form)

This command (no query form) performs the 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

This command forces a trigger event to occur. The query returns the current trigger parameters for the instrument.

Group Trigger

Syntax

```
TRIGger FORCe
TRIGger?
```

Arguments `FORCe` creates a trigger event. If TRIGger:STATE is set to READy, the acquisition will complete. Otherwise, this command will be ignored. This is equivalent to selecting A Event (Main) Trigger Setup from the Trig menu and then clicking Force Trigger.

Examples `TRIGGER FORCE` forces a trigger event to occur.

`TRIGGER?` might return these trigger parameters:

```
:TRIGGER:A:MODE AUTO;TYPE EDGE;LEVEL 0.0000;HOLDOFF:BY DEFAULT;TIME
1.5000E-6;:TRIGGER:A:EDGE:SOURCE CH1;COUPLING
DC;SLOPE RISE;:TRIGGER:A:LOGIC:CLASS PATTERN;FUNCTION
AND;THRESHOLD:CH1 1.2000;CH2 1.2000;CH3 1.2000;CH4
1.2000;:TRIGGER:A:LOGIC:INPUT:CH1 HIGH;CH2
X;CH3 X;:TRIGGER:A:LOGIC:PATTERN:INPUT:CH4
X;:TRIGGER:A:LOGIC:PATTERN:WHEN
TRUE;WHEN:LESSLIMIT 5.0000E-9;MORELIMIT
5.0000E-9;:TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE RISE;THRESHOLD
1.2000;SOURCE CH2;:TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD
```

```
1.2000;SOURCE CH1;:TRIGGER:A:LOGIC:SETHOLD:HOLDTIME
2.0000E-9;SETTIME 3.0000E-9;:TRIGGER:A:LOGIC:STATE:INPUT:CH4
RISE;:TRIGGER:A:LOGIC:STATE:WHEN
TRUE;:TRIGGER:A:PULSE:CLASS GLITCH;SOURCE
CH1;GLITCH:WIDTH 2.0000E-9;TRIGIF ACCEPT;POLARITY
POSITIVE;:TRIGGER:A:PULSE:WINDOW:TYPE INSIDE;WHEN
OCCURS;WIDTH 2.0000E-9;LOGIC:INPUT:CH1 HIGH;CH2 HIGH;CH3
HIGH;CH4 HIGH;:TRIGGER:A:PULSE:WINDOW:LOGIC:THRESHOLD:CH1
1.2000;CH2 1.2000;CH3 1.2000;CH4
1.2000;:TRIGGER:A:PULSE:WINDOW:THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:RUNT:LOGIC:THRESHOLD:CH1
1.2000;CH2 1.2000;CH3 1.2000;CH4
1.2000;:TRIGGER:A:PULSE:RUNT:POLARITY
POSITIVE;THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:RUNT:WHEN OCCURS;WIDTH
2.0000E-9;:TRIGGER:A:PULSE:TRANSITION:DELTATIME
2.0000E-9;POLARITY POSITIVE;THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:TRANSITION:WHEN
FASTERTHAN;:TRIGGER:A:PULSE:WIDTH:LOWLIMIT
2.0000E-9;HIGHLIMIT 2.0000E-9;WHEN WITHIN;POLARITY
POSITIVE;:TRIGGER:A:PULSE:TIMEOUT:POLARITY
STAYSHIGH;TIME 2.0000E-9;:TRIGGER:A:VIDEO:CUSTOM:FORMAT
INTERLACED;SCAN RATE1;:TRIGGER:A:VIDEO:FIELD
ALLFIELDS;HOLDOFF:FIELD 0.0000;:TRIGGER:A:VIDEO:LINE
1;POLARITY NORMAL;SCAN RATE1;SOURCE CH1;STANDARD
NTSC;:TRIGGER:A:COMMUNICATION:STANDARD CUSTOM;CODE
NRZ;BITRATE 1544000;AMI:PULSEFORM PLUSONE;THRESHOLD:HIGH
0.0000;LOW 0.0000;:TRIGGER:A:COMMUNICATION:CLOCK:POLARITY
RISE;:TRIGGER:A:COMMUNICATION:CMI:PULSEFORM
PLUSONE;:TRIGGER:A:COMMUNICATION:SOURCE CH1;SOURCE:TYPE
DATA;:TRIGGER:B:STATE 0;TYPE EDGE;LEVEL 0.0000;BY
EVENTS;EDGE:SOURCE CH1;SLOPE RISE;COUPLING
DC;:TRIGGER:B:TIME 16.0000E-9;EVENTS:COUNT 2.
```

TRIGger:{A|B}:EDGE? (Query Only)

This query-only command returns the trigger source, coupling, and slope for the specified edge trigger. This command is equivalent to selecting Edge Setup from the Trig menu and viewing the current setups, or selecting B Event (Delayed) Trigger Setup from the Trig menu and viewing the current Source, Slope, and Coupling settings.

Group Trigger

Syntax `TRIGger:{A|B}:EDGE?`

Related Commands TRIGger:{A|B}:PULse?, TRIGger:{A|B}:LOGIc?, TRIGger:{A|B}:EDGE:COUPling, TRIGger:{A|B}:EDGE:SLOpe, TRIGger:{A|B}:EDGE:SOUrce, TRIGger:B:STATE

Examples `TRIGGER:A:EDGE?` might return `:TRIGGER:A:EDGE:SOURCE CH1;COUPLING DC; SLOPE RISE`, indicating the trigger source, coupling, and slope for the A edge trigger.

TRIGger:{A|B}:EDGE:COUPling

This command sets or queries the type of coupling for the edge trigger. This command is equivalent to selecting Event Trigger Setup from the Trig menu, selecting Edge Trigger, and choosing from the Coupling drop-down list.

Group Trigger

Syntax

```
TRIGger:{A|B}:EDGE:COUPling
{AC|DC|HFRej|LFRej|NOISErej|ATRIGger}
TRIGger:{A|B}:EDGE:COUPling?
```

Related Commands TRIGger:{A|B}:EDGE:SOUrce, TRIGger:{A|B}:EDGE:SLOpe

Arguments

`AC` selects AC trigger coupling, which passes the input signals above 60 Hz to the trigger circuitry.

`DC` selects DC trigger coupling, which passes all input signals to the trigger circuitry.

`HFRej` coupling attenuates signals above 50 kHz before passing the signals to the trigger circuitry.

`LFRej` coupling attenuates signals below 80 kHz before passing the signals to the trigger circuitry.

`NOISErej` coupling provides stable triggering by increasing the trigger hysteresis. Increased hysteresis reduces the trigger sensitivity to noise but can require greater trigger signal amplitude.

`ATRIGger` this B trigger command sets the B trigger coupling to match the setting on the A trigger.

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|B}:EDGE:COUPling:CH<x>

This command sets or queries the type of coupling for the A or B trigger for the specified channel. This command is equivalent to selecting A or B Trigger Setup from the Trig menu and choosing the setting from the Coupling drop-down list.

Group Trigger

Syntax
```
TRIGger:{A|B}:EDGE:COUPling:CH<x>
{AC|DC|HFRej|LFRej|NOISErej}
TRIGger:{A|B}:EDGE:COUPling:CH<x>?
```

Related Commands TRIGger:{A|B}:EDGE:COUPling

Arguments

`AC` selects AC trigger coupling.

`DC` selects `DC` trigger coupling.

`HFRej` selects high frequency low sensitivity.

`LFRej` selects low frequency low sensitivity.

`NOISErej` selects `DC` low sensitivity.

Examples

`TRIGGER:B:EDGE:COUPLING:CH1 DC` selects DC for the B trigger coupling of Channel 1.

`TRIGGER:B:EDGE:COUPLING:CH1?` might return `:TRIGGER:B:EDGE:COUPLING:CH1 LFREJ` for the B trigger coupling.

TRIGger:{A|B}:EDGE:SLOpe

This command sets or queries the slope for the edge trigger. This command is equivalent to selecting Event Trigger Setup from the Trig menu and then choosing the desired Slope.

Group Trigger

Syntax
```
TRIGger:{A|B}:EDGE:SLOpe {RISe|FALL|EITher}
TRIGger:{A|B}:EDGE:SLOpe?
```

Related Commands TRIGger:{A|B}:EDGE:SOUrce, TRIGger:{A|B}:EDGE:COUPling, TRIGger:{A|B}:EDGE?, TRIGger:B:STATE

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 specified to trigger on either the rising or falling edge of a signal.

NOTE. *The edge trigger slope for channels 1-4 and Aux can be RISE, FALL, or EITHER on the DPO7000C and DPO/DSA/MSO70000C/D/DX Series instruments. For MSO/DPO5000/B oscilloscopes, the edge trigger slope for channels 1-4 can be RISE or FALL.*

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|B}:EDGE:SLOpe:AUX

This command sets or queries the slope for the edge trigger for the AUX In input. This command is equivalent to selecting A Event or B Event Trigger Setup from the Trig menu and then choosing the desired Slope.

Group

Trigger

Syntax

```
TRIGger:{A|B}:EDGE:SLOpe:AUX {RISe|FALL|EITHER}
TRIGger:{A|B}:EDGE:SLOpe:AUX?
```

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 on either the rising or falling edge of a signal. EITHER is only available on the DPO7000C and DPO/DSA/MSO70000C/D/DX Series instruments.

Examples

TRIGGER:A:EDGE:SLOPE:AUX RISE sets the A edge trigger slope to positive, which triggers on the rising edge of the signal.

TRIGGER:A:EDGE:SLOPE:AUX? might return :TRIGGER:A:EDGE:SLOPE:AUX FALL, indicating that the A edge trigger slope is negative.

TRIGger:{A|B}:EDGE:SLOpe:CH<x>

This command sets or queries the slope for the edge trigger for the specified channel. This command is equivalent to selecting A Event or B Event Trigger Setup from the Trig menu and then choosing the desired Slope.

Group Trigger

Syntax

```
TRIGger:{A|B}:EDGE:SLOpe:CH<x> {RISe|FALL|EITher}
TRIGger:{A|B}:EDGE:SLOpe:CH<x>?
```

Related Commands TRIGger:{A|B}:EDGE:SOUrce, TRIGger:{A|B}: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.

`EITher` specifies to trigger on either the rising or falling edge of a signal. `EITHER` is only available on the DPO7000C and DPO/DSA/MSO70000C/D/DX Series instruments.

Examples

`TRIGGER:A:EDGE:SLOPE:CH2 RISE` sets the Channel 2 edge trigger slope to rising or positive

`TRIGGER:A:EDGE:SLOPE:CH1?` might return `:TRIGGER:A:EDGE:SLOPE FALL`, indicating that the A edge trigger slope is negative.

TRIGger:{A|B}:EDGE:SOUrce

This command sets or queries the source for the edge trigger. This command is equivalent to selecting Event Trigger Setup from the Trig menu and then choosing from the Source drop-down list.

Group Trigger

Syntax

```
TRIGger:{A|B}:EDGE:SOUrce {AUXiliary|CH<x>|LINE|D<x>}
TRIGger:{A|B}:EDGE:SOUrce?
```

Related Commands TRIGger:{A|B}:EDGE:SLOpe, TRIGger:{A|B}:EDGE:COUPling, TRIGger:{A|B}:EDGE?, TRIGger:B:STATE

Arguments

AUXiliary specifies an external trigger using the Auxiliary Trigger Input. x can be 1, 2, 3, or 4.

CH<x> specifies one input channel as the edge trigger source.

LINE specifies AC line voltage, and is for A Trigger only.

D<x> specifies a digital input as the edge trigger source. x can be 0 through 15.

Examples

TRIGGER:A:EDGE:SOURCE CH2 sets the A edge trigger source to input channel CH2.

TRIGGER:A:EDGE:SOURCE? might return :TRIGGER:A:EDGE:SOURCE CH1, indicating that the A edge trigger source is set to input channel CH1.

TRIGger:{A|B}:LEVel

This command sets or queries the level for the trigger. This command is equivalent to selecting Holdoff from the Trig menu and then viewing or setting the trigger Level or selecting B Event (Delayed) Trigger Setup from the Trig menu and setting the B Trig Level voltage.

Group

Trigger

Syntax

```
TRIGger:{A|B}:LEVel {ECL|TTL|<NR3>}
TRIGger:{A|B}:LEVel?
```

Related Commands

TRIGger:{A|B}:LEVel, TRIGger:B, TRIGger:{A|B}:EDGE:SOUrce
TRIGger:B:STATE

Arguments

ECL specifies the ECL high level.

TTL specifies the TTL high level.

<NR3> specifies the trigger level in user units (usually volts).

Examples

TRIGGER:A:LEVEL TTL sets the A edge trigger to TTL high level.

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|B}:LEVel:CH<x>

This command sets or queries the CH<x> trigger level for TRIGGER:LVLSRCPREFERENCE SRCDEPENDENT mode.

The CH<x> range is 1 to 4.

Group Trigger

Syntax
```
TRIGger:{A|B}:LEVel:CH<x> {ECL|TTL|<NR3>}
TRIGger:{A|B}:LEVel:CH<x>?
```

Arguments ECL specifies the ECL high level.

TTL specifies the TTL high level.

<NR3> specifies the trigger level in user units (usually volts).

Examples TRIGGER:A:LEVEL:CH1 TTL sets the A trigger level for Channel 1 to the TTL high level.

TRIGGER:A:LEVEL:CH2? might return :TRIGGER:A:LEVEL:CH2 1.3000E+00, indicating that the A trigger level for Channel 2 is set to 1.3 V.

TRIGger:{A|B}:LOGIc? (Query Only)

This query-only command returns all of the logic trigger parameters.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc?
```

Related Commands TRIGger:{A|B}:LOGIc:CLAss

Examples TRIGGER:A:LOGIC? might return
```
:TRIGGER:A:LOGIC:CLASS PATTERN; FUNCTION AND;WHEN TRUE; THRESHOLD:CH1 1.4000;CH2 1.4000;CH3 1.4000; CH4 1.4000;:TRIGGER:A:LOGIC:INPUT:CH1 HIGH; CH2 X;CH3 X; :TRIGGER:A:LOGIC:PATTERN:INPUT:CH4 X; :TRIGGER:A:LOGIC:PATTERN:WHEN TRUE; WHEN:LESSLIMIT 5.0000E-9; MORELIMIT
```

```
5.0000E-9; :TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE
RISE; THRESHOLD 1.4000;SOURCE CH2;
:TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD 1.4000;SOURCE
CH1; :TRIGGER:A:LOGIC:SETHOLD:HOLDTIME2.0000E-9; SETTIME
3.0000E-9; :TRIGGER:A:LOGIC:STATE:INPUT:CH4 RISE;
:TRIGGER:A:LOGIC:STATE:WHEN TRUE.
```

TRIGger:{A|B}:LOGIc:CLAss

NOTE. *Not available on some models.*

This command sets or queries the class of the Logic Trigger. Used with the TRIGger:{A|B}:TYPe command, this command is equivalent to selecting Logic Pattern, Logic State, or Setup/Hold Setup from the Trig menu.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:CLAss {PATtern|STATE|SETHold}
TRIGger:{A|B}:LOGIc:CLAss?
```

Related Commands TRIGger:{A|B}:TYPe, TRIGger:{A|B}:PULse:CLAss

Arguments

`PATtern` sets the instrument to trigger when the specified logical combinations of Channels 1, 2, 3, and 4 are met.

`STATE` sets the instrument to trigger when the specified conditions of Channels 1, 2, and 3 are met after the Channel 4 (clock) condition is met.

`SETHold` sets the instrument 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 PATTERN` sets the trigger A logic class to PATtern, which causes the instrument to trigger when the specified logical combinations of Channels 1, 2, 3, and 4 are met.

`TRIGGER:A:LOGIC:CLASS?` might return `:TRIGGER:A:LOGIC:CLASS PATTERN`, indicating that the A trigger level for channel 2 is set to PATTERN, which causes the instrument to trigger when the specified logical combinations of Channels 1, 2, 3, and 4 are met.

TRIGger:{A|B}:LOGIc:FUNCtion

NOTE. *Not available on some models.*

This command sets or queries the logical combination of the input channels for the pattern and state logic triggers. This command is equivalent to selecting Event Trigger Setup from the Trig menu, selecting Pattern or State for the Trigger Type, and setting or viewing the Pattern Type.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:FUNCtion {AND|NANd|NOR|OR}
TRIGger:{A|B}:LOGIc:FUNCtion?
```

Related Commands TRIGger:{A|B}:LOGIc:PATtern:WHEn, TRIGger:{A|B}:LOGIc:INPut:CH<x>, TRIGger:{A|B}:LOGIc:THReshold:CH<x>

Arguments

`AND` specifies to trigger if all conditions are true.

`NANd` specifies to trigger if any of the conditions are false.

`NOR` specifies to trigger if all conditions are false.

`OR` specifies to trigger if any of the conditions are true.

Examples

`TRIGGER:A:LOGIC:FUNCTION AND` sets the logical combination of channels to be true when all conditions are true.

`TRIGGER:A:LOGIC:FUNCTION?` might return `:TRIGGER:A:LOGIC:FUNCTION NAND`, indicating that the instrument will trigger if the AND logic conditions are false.

TRIGger:{A|B}:LOGIc:INPut? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the logic trigger input expected for Channel 1, 2, and 3. Channel 4 is set or queried with the command TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>.

This command is equivalent to selecting Event Trigger Setup and viewing or setting the Input Threshold for the channels.

Group Trigger

Syntax `TRIGger:{A|B}:LOGIc:INPut?`

Related Commands TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>

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 3 are “don't care.”

TRIGger:{A|B}:LOGIc:INPut:ALL

NOTE. *Not available on some models.*

This command sets or queries the A or B logic trigger input condition for all the channels. The command is available when the Trigger Type is set to Pattern/State.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:INPut:ALL <QString>
TRIGger:{A|B}:LOGIc:INPut:ALL?
```

Related Commands TRIGger:{A|B}:LOGIc:INPut:FORMat

Arguments `<QString>` specifies the bit pattern for all the channels.

Examples `TRIGGER:A:LOGIC:INPUT:ALL “0xxxx”` sets logic high to channel D0 and “don’t care” condition for CH1 through CH4.

`TRIGGER:A:LOGIC:INPUT:ALL?` might return `:TRIGGER:A:LOGIC:INPUT:ALL “1111xxxx”`, indicating that a logic high is set on D3 through D0 and a “don’t care” condition is set on CH1 through CH4.

TRIGger:{A|B}:LOGIc:INPut:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the A or B logical trigger input for the channel specified by x. The value of x ranges from 1 through 3. Note that CH4 cannot be set or queried with this command. For details about setting this channel, see TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>. This command is equivalent to selecting Event Trigger Setup from the Trig menu and then choosing the desired logical input from the Ch<x> drop-down list, which is located in the Input Threshold group box.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:INPut:CH<x> {HIGH|LOW|X}
TRIGger:{A|B}:LOGIc:INPut:CH<x>?
```

Related Commands TRIGger:{A|B}:LOGIc:PATtern: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:CH2 HIGH` sets the A logic trigger input to logic HIGH for Channel 2.

`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|B}:LOGIc:INPut:FORMat

NOTE. *Not available on some models.*

This command sets or queries the A or B logic trigger pattern format.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:INPut:FORMat {HEXadecimal|BINary}
TRIGger:{A|B}:LOGIc:INPut:FORMat?
```

Related Commands TRIGger:{A|B}:LOGIc:INPut:ALL

Arguments

`HEXadecimal` specifies hexadecimal pattern format for A or B logic trigger.

`BINary` specifies binary pattern format for A or B logic trigger.

Examples

`TRIGGER:A:LOGIC:INPUT:FORMAT HEX` sets the A logic trigger pattern format to hexadecimal.

`TRIGGER:A:LOGIC:INPUT:FORMAT?` might return `:TRIGGER:A:LOGIC:INPUT:FORMAT BINARY`, indicating that the A logic trigger pattern format is set to binary.

TRIGger:{A|B}:LOGIc:PATtern? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the conditions used for generating an A logic pattern trigger, with respect to the defined input pattern, and identifies the maximum and minimum time that the selected pattern can be true and still generate the trigger. This command is equivalent to selecting Logic Pattern from the Trig menu and then viewing the current setups.

Group

Trigger

Syntax

`TRIGger:{A|B}:LOGIc:PATtern?`

Related Commands

TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>, TRIGger:{A|B}:LOGIc:PATtern:WHEn, TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit, TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit

Examples

`TRIGGER:A:LOGIC:PATTERN?` might return
`:TRIGGER:A:LOGIC:PATTERN:INPUT:CH4 X;`
`:TRIGGER:A:LOGIC:PATTERN:WHEN TRUE; WHEN:LESSLIMIT 5.0000E-9;MORELIMIT 5.0000E-9.`

TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the A or B logic trigger input for the specified channel. This command specifies the logic value used when the pattern trigger detects the threshold level.

This command is equivalent to selecting Logic Pattern from the Trig menu and then choosing the desired logical input from the channel drop-down list, which is located in the Input Threshold group box.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x> {HIGH|LOW|X}
TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>?
```

Related Commands TRIGger:{A|B}:LOGIc:FUNCtion, TRIGger:{A|B}:LOGIc:INPut:CH<x>, TRIGger:{A|B}:LOGIc:THReshold:CH<x>

Arguments

`HIGH` specifies the logic high.

`LOW` specifies the logic low.

`X` specifies a “don't care” state.

Examples

`TRIGGER:A:LOGIC:PATTERN:INPUT:CH4 HIGH` This command sets the A logic trigger input to logic high for Channel 4 when the logic class is set to PATtern. When the threshold level is detected, High places a 1 on the Channel 4 input to the selected function.

`TRIGGER:A:LOGIC:PATTERN:INPUT:CH4?` might return `:TRIGGER:A:LOGIC:PATTERN:INPUT:CH4 HIGH`, indicating that the logic input for Channel 4 is logic high.

TRIGger:{A|B}:LOGIc:PATtern:WHEn

NOTE. *Not available on some models.*

This command sets or queries the condition for generating an A or B logic pattern trigger with respect to the defined input pattern. This command is equivalent to selecting A or B Event (Main) Trigger Setup from the Trig menu, selecting Pattern for Trigger Type, and choosing a trigger condition from the Pattern drop-down list, which is located in the Trigger When group box.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:PATtern:WHEn
{TRUe|FALSe|LESSThan|MOREThan}
TRIGger:{A|B}:LOGIc:PATtern:WHEn?
```

Related Commands

TRIGger:{A|B}:LOGIc:FUNCtion, TRIGger:{A|B}:LOGIc:INPut?, TRIGger:{A|B}:LOGIc:THReshold?, TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit, TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit

Arguments

`TRUe` sets the instrument to trigger when the pattern becomes true.

`FALSe` sets the instrument to trigger when the pattern becomes false.

`LESSThan` sets the instrument to trigger if the specific pattern is true less than the time set by the TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit command.

`MOREThan` argument sets the instrument to trigger if the specific pattern is true longer than the specified time set by the TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit command.

Examples

`TRIGGER:A:LOGIC:PATTERN:WHEN FALSE` specifies to trigger the A logic pattern when the pattern becomes false.

`TRIGGER:A:LOGIC:PATTERN:WHEN?` might return `:TRIGGER:A:LOGIC:PATTERN:WHEN TRUE`, indicating that the A logic pattern will trigger when the pattern becomes true.

TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit

NOTE. *Not available on some models.*

This command sets or queries the maximum time that the selected pattern can be true and still generate an A or B logic pattern trigger. This command is equivalent to selecting the A or B Event (Main) Trigger Setup from the Trig menu, selecting Pattern as the Trigger Type, selecting Less Than for the Pattern in the Trigger When settings, and entering a maximum value for Time.

Group

Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit <NR3>
TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit?
```

Related Commands

TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit

Arguments <NR3> specifies the amount of time to hold the pattern true.

Examples TRIGGER:A:LOGIC:PATTERN:WHEN:LESSLIMIT 10.0E+00 sets the maximum time that the selected pattern can hold true (and generate an A logic pattern trigger) to 10 s.

TRIGGER:A:LOGIC:PATTERN:WHEN:LESSLIMIT? might return :TRIGGER:A:LOGIC:PATTERN:WHEN:LESSLIMIT 5.0000E-09, indicating that the selected pattern can hold true for up to 5 ns and still generate an A logic pattern trigger.

TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit

NOTE. *Not available on some models.*

This command sets or queries the minimum time that the selected pattern can be true and still generate an A or B logic pattern trigger. This command is equivalent to selecting A or B Event (Main) Trigger Setup from the Trig menu, selecting Pattern as the Trigger Type, selecting More Than for the Pattern in the Trigger When settings, and entering a minimum value for Time.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit <NR3>
TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit?
```

Related Commands TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit

Arguments <NR3> specifies the amount of time to hold the pattern true.

Examples TRIGGER:A:LOGIC:PATTERN:WHEN:MORELIMIT 10.0E+00 sets the minimum time that the selected pattern can hold true (and generate an A logic pattern trigger) to 10 s.

TRIGGER:A:LOGIC:PATTERN:WHEN:MORELIMIT? might return :TRIGGER:A:LOGIC:PATTERN:WHEN:MORELIMIT 5.0000E-09, indicating that the selected pattern must hold true for at least 5 ns to generate an A logic pattern trigger.

TRIGger:{A|B}:LOGIc:SETHold? (Query Only)

NOTE. *Not available on some models.*

This query-only command 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. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and then viewing the current setups.

Group Trigger

Syntax `TRIGger:{A|B}:LOGIc:SETHold?`

Related Commands TRIGger:{A|B}:LOGIc:PATtern?, TRIGger:{A|B}:LOGIc:STATE?

Examples `TRIGGER:A:LOGIC:SETHOLD?` might return the settings
`:TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE RISE; THRESHOLD 1.4000;THRESHOLD:CH1 1.4000;CH2 1.2000;CH3 1.2000;CH4 1.2000;:TRIGGER:A:LOGIC:SETHOLD:CLOCK:SOURCE CH2;:TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD 1.4000;THRESHOLD: CH1 1.4000;CH2 1.2000;CH3 1.2000;CH4 1.2000;:TRIGGER:A:LOGIC:SETHOLD :HOLDTIME 2.0000E-9;SETTIME 3.0000E-9;QUALIFY OCCURS`.

TRIGger:{A|B}:LOGIc:SETHold:CLOCk? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the clock edge polarity, voltage threshold and source input for setup and hold triggering. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and then viewing the current clock setups.

Group Trigger

Syntax `TRIGger:{A|B}:LOGIc:SETHold:CLOCk?`

Related Commands TRIGger:{A|B}:LOGIc:SETHold:DATa?, TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE, TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold, TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce

Examples `TRIGGER:A:LOGIC:SETHOLD:CLOCK?` might return `:TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE RISE;THRESHOLD 1.2000;THRESHOLD:CH1 1.2000;CH2 1.2000; CH3 1.2000;CH4 1.2000;:TRIGGER:A:LOGIC:SETHOLD:CLOCK:SOURCE CH2`, indicating the current clock settings for setup and hold triggering.

TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE

NOTE. *Not available on some models.*

This command sets or queries the clock edge polarity for setup and hold triggering. This is equivalent to selecting Setup/Hold Setup from the Trig menu and then choosing the desired Clock Edge.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE {FALL|RISe}
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE?
```

Related Commands TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce, TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold

Arguments `FALL` specifies polarity as the clock falling edge.

`RISe` specifies polarity as the clock rising edge.

Examples `TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE RISE` specifies the polarity as the clock rising edge.

`TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE?` might return `:TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE RISE`, indicating that polarity is specified as the clock rising edge.

TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the clock source for the A or B logic trigger setup and hold input. This is equivalent to selecting Setup/Hold Setup from the Trig menu and choosing the desired channel from the Clock Source drop-down list.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce CH<x>
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce?
```

Related Commands TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE, TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold

Arguments `CH<x>` specifies the input channel, which ranges from 1 through 4 for four-channel instruments or 1 through 2 for two channel instruments.

Examples `TRIGGER:A:LOGIC:SETHOLD:CLOCK:SOURCE CH1` specifies channel 1 as the A logic setup and hold input.

`TRIGGER:A:LOGIC:SETHOLD:CLOCK:SOURCE?` might return `:TRIGGER:A:LOGIC:SETHOLD:CLOCK:SOURCE CH4`, indicating that channel 4 is the clock source for the setup and hold input for the A logic trigger.

TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold

NOTE. *Not available on some models.*

This command sets or queries the clock voltage threshold for the setup and hold trigger. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and setting the desired Clock Level.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold {ECL|TTL|<NR3>}
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold?
```

Related Commands TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE, TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce

Arguments

ECL specifies a preset ECL high level.

TTL specifies a preset TTL high level.

<NR3> is the clock level in volts.

Examples

TRIGGER:A:LOGIC:SETHOLD:CLOCK:THRESHOLD TTL specifies the preset TTL value as the clock threshold for the setup and hold trigger.

TRIGGER:A:LOGIC:SETHOLD:CLOCK:THRESHOLD? might return :TRIGGER:A:LOGIC:SETHOLD:CLOCK:THRESHOLD 1.2000E+00, indicating that the clock threshold for the setup and hold trigger is 1.2 V.

TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the clock voltage threshold for setup and hold trigger. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and then setting the desired Clock Level. The value of x can range from 1 through 4.

Group

Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold:CH<x>
{ECL|TTL|<NR3>}
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold:CH<x>?
```

Related Commands

TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE, TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce

Arguments

ECL specifies a preset ECL high level.

TTL specifies a preset TTL high level.

<NR3> is the clock level in volts.

Examples

TRIGGER:A:LOGIC:SETHOLD:CLOCK:THRESHOLD:CH3 TTL specifies the preset TTL value as the clock threshold for setup and hold trigger for Channel 3.

TRIGGER:A:LOGIC:SETHOLD:CLOCK:THRESHOLD:CH3? might return :TRIGGER:A:LOGIC:SETHOLD:CLOCK:THRESHOLD:CH3 1.2000E+00, indicating that the clock threshold for setup and hold trigger is 1.2 V for Channel 3.

TRIGger:{A|B}:LOGIc:SETHold:DATa? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the voltage threshold and data source for the setup and hold trigger. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and then viewing the current data setups.

Group Trigger

Syntax `TRIGger:{A|B}:LOGIc:SETHold:DATa?`

Related Commands TRIGger:{A|B}:LOGIc:SETHold:CLOCk?

Examples `TRIGGER:A:LOGIC:SETHOLD:DATA?` might return `:TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE RISE;THRESHOLD 1.2000;THRESHOLD:CH1 1.2000;CH2 1.2000; CH3 1.2000;CH4 1.2000;:TRIGGER:A:LOGIC:SETHOLD:CLOCK:SOURCE CH2`, indicating the current trigger data settings.

TRIGger:{A|B}:LOGIc:SETHold:DATa:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the data source for the setup and hold trigger. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and choosing the desired channel from the Data Source drop-down list.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:SETHold:DATa:SOUrce CH<x>
TRIGger:{A|B}:LOGIc:SETHold:DATa:SOUrce?
```

Related Commands TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold

Arguments `CH<x>` specifies the input channel, which ranges from 1 through 4.

Examples

`TRIGGER:A:LOGIC:SETHOLD:DATA:SOURCE CH1` sets channel 1 as the clock source for the setup and hold trigger.

`TRIGGER:A:LOGIC: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|B}:LOGIc:SETHold:DATa:THReshold

NOTE. *Not available on some models.*

This command sets or queries the data voltage threshold for setup and hold trigger. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and then setting the desired Data Level.

Group

Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold {ECL|TTL|<NR3>}
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold?
```

Related Commands

TRIGger:{A|B}:LOGIc:SETHold:DATa:SOUrce

Arguments

`ECL` specifies the preset `ECL` high level.

`TTL` specifies the preset TTL high level.

`<NR3>` is the setup and hold data level in V.

Examples

`TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD TTL` specifies the preset TTL high level as the current data voltage level for the setup and hold trigger.

`TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD?` might return `:TRIGGER:A:LOGIC: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|B}:LOGIc:SETHold:DATa:THReshold:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the data voltage threshold for setup and hold trigger.

This command is equivalent to selecting A or B Event Trigger Setup from the Trig menu and then setting the desired Data Level. The value of x can range from 1 through 4.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold:CH<x>
{ECL|TTL|<NR3>}
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold:CH<x>?
```

Related Commands TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold:CH<x>

Arguments

`ECL` specifies the preset ECL high level.

`TTL` specifies the preset TTL high level.

`<NR3>` is the setup and hold data level in V.

Examples

`TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD:CH2 TTL` specifies the preset TTL high level as the current data voltage level for setup and hold trigger for Channel 2.

`TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD:CH2?` might return `:TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD:CH2 1.2000E+00`, indicating that 1.2 V is the current data voltage level for setup and hold trigger for Channel 2.

TRIGger:{A|B}:LOGIc:SETHold:HOLDTime

NOTE. *Not available on some models.*

This command sets or queries the hold time for setup and hold violation triggering. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and then setting the desired Hold Time.

Group Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:SETHold:HOLDTime <NR3>
TRIGger:{A|B}:LOGIc:SETHold:HOLDTime?
```

Related Commands TRIGger:{A|B}:LOGIc:SETHold:SETTime

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:LOGIC:SETHOLD:HOLDTIME 3.0E-3 sets the hold time for the setup and hold trigger to 3 ms.

TRIGGER:A:LOGIC:SETHOLD:HOLDTIME? might return :TRIGGER:A:LOGIC:SETHOLD:HOLDTIME 2.0000E-09, indicating that the current hold time for the setup and hold trigger is 2 ns.

TRIGger:{A|B}:LOGIc:SETHold:QUAlify

NOTE. *Not available on some models.*

This command sets or queries the Setup/Hold Trigger qualification. This is equivalent to selecting Setup/Hold Setup from the Trig menu and selecting Occurs, Logic, or Bus in the Trigger If Setup/Hold drop-down list box.

NOTE. *If you use the LOGIc argument, you must specify the channel logic conditions.*

BUS is same as OccursAnd in the user interface.

NOTE. *In Trigger B logic setup/hold, the OccursAnd option is not available in the Trigger If Setup/Hold drop-down list box.*

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:SETHold:QUAlify {OCCurs|LOGIc|BUS}
TRIGger:{A|B}:LOGIc:SETHold:QUAlify?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments

`OCCurs` specifies a trigger if any detectable event occurs.

`LOGIc` specifies a trigger if the individual channel qualifications meet the logic patterns and thresholds set by the TRIGger:{A|B}:LOGIc:INPut:CH<x> and TRIGger:{A|B}:LOGIc:THReshold:CH<x> commands.

`BUS` specifies a trigger if the bus input qualifications meet the pattern set by the TRIGger:QUALification:BUS:VALue command.

Examples

`TRIGGER:A:LOGIC:SETHOLD:QUALIFY OCCURS` sets the trigger qualifier to OCCurs, which initiates a trigger if the instrument detects any setup and hold trigger qualifications.

`TRIGGER:A:LOGIC:SETHOLD:QUALIFY?` might return `:TRIGGER:A:LOGIC:SETHOLD:QUALIFY OCCURS`, indicating that a trigger will occur if the instrument detects any of the setup and hold qualifications.

TRIGger:{A|B}:LOGIc:SETHold:SETTime

NOTE. *Not available on some models.*

This command sets or queries the setup time for setup and hold violation triggering. This command is equivalent to selecting Setup/Hold Setup from the Trig menu and then setting the desired Setup Time.

Group

Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:SETHold:SETTime <NR3>
TRIGger:{A|B}:LOGIc:SETHold:SETTime?
```

Related Commands

TRIGger:{A|B}:LOGIc:SETHold:HOLDTime

Arguments

`<NR3>` specifies the setup time for setup and hold violation triggering.

Examples

`TRIGGER:A:LOGIC:SETHOLD:SETTIME 3.0E-6` specifies that the current setup time for setup and hold trigger is 3 µs.

`TRIGGER:A:LOGIC: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|B}:LOGIc:STATE? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the data input and trigger criteria for the logic trigger. This command is equivalent to selecting Logic State from the Trig menu and then viewing the current logic state settings.

Group Trigger

Syntax `TRIGger:{A|B}:LOGIc:STATE?`

Related Commands TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x>, TRIGger:{A|B}:LOGIc:STATE:WHEn

Examples `TRIGGER:A:LOGIC:STATE?` might return
`:TRIGGER:A:LOGIC:STATE:INPUT:CH4 RISE;:TRIGGER:A:LOGIC:STATE:WHEN TRUE;CLOCK:SOURCE CH4`.

TRIGger:{A|B}:LOGIc:STATE:CLOck:SOUrce

NOTE. *Not available on some models. This command is valid only on MSO instruments. DPO instruments do not support it.*

This command sets or queries the clock source for the A or B logic trigger.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:STATE:CLOck:SOUrce {CH4|CQ}
TRIGger:{A|B}:LOGIc:STATE:CLOck:SOUrce?
```

NOTE. *`CQ` is available only on MSO Series of products.*

Arguments `CH4` specifies the analog channel as clock source.

`CQ` specifies the channel/qualifier as clock source.

Examples

`TRIGGER:A:LOGIC:STATE:CLOCK:SOURCE CQ` sets the clock source to channel/qualifier.

`TRIGGER:A:LOGIC:STATE:CLOCK:SOURCE?` might return `:TRIGGER:A:LOGIC:STATE:CLOCK:SOURCE CH4`, indicating that the clock source is set to CH4.

TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the slope for the channel specified by x when the logic class is set to State. This command is equivalent to selecting Logic State from the Trig menu, choosing the desired channel input, and then the slope (NEG or POS) from the When Clock is drop-down list.

Group

Trigger

Syntax

```
TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x> {FALL|RISe}
TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x>?
```

Related Commands

TRIGger:{A|B}:LOGIc:STATE:WHEn

Arguments

`FALL` specifies the falling edge and the input slope is NEG.

`RISe` specifies the rising edge and the input slope is POS.

Examples

`TRIGGER:A:LOGIC:STATE:INPUT:CH4 RISE` specifies that the A logic trigger input for Channel 4 is the rising edge.

`TRIGGER:A:LOGIC:STATE:INPUT:CH4?` might return `:TRIGGER:A:LOGIC:STATE:INPUT:CH4 RISE`, indicating that the A logic trigger input for Channel 4 is the rising edge.

TRIGger:{A|B}:LOGIc:STATE:WHEn

NOTE. *Not available on some models.*

This command sets or queries the condition for generating an A or B logic state trigger. This command is equivalent to selecting Logic State from the Trig menu and choosing the desired condition from the Trigger When Pattern drop-down list.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:STATE:WHEn {TRUe|FALSe}
TRIGger:{A|B}:LOGIc:STATE:WHEn?
```

Related Commands TRIGger:{A|B}:LOGIc:INPut:CH<x>, TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x>

Arguments `TRUe` specifies that the trigger occurs when the clock transition on channel 4 occurs and the pattern of channels 1-3 are at the desired logic input states.

`FALSe` specifies that the trigger occurs when the desired clock transition on channel 4 occurs and the desired logic input states on channels 1-3 are not found.

Examples `TRIGGER:A:LOGIC:STATE:WHEN TRUE` specifies that the logic condition to trigger upon is true.

`TRIGGER:A:LOGIC:STATE:WHEN?` might return `:TRIGGER:A:LOGIC:STATE:WHEN FALSE`, indicating that the logic condition to trigger upon is false.

TRIGger:{A|B}:LOGIc:THReshold? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the threshold voltage for all channels in a logic trigger. This command query is equivalent to selecting Event Trigger Setup from the Trig menu, choosing a logic trigger type, such as State or Pattern, and viewing the current Input Threshold voltage settings.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:THReshold?
```

Related Commands TRIGger:{A|B}:LOGIc:THReshold:CH<x>, TRIGger:{A|B}:LOGIc:INPut?, TRIGger:{A|B}:LOGIc:INPut:CH<x>

Examples `TRIGGER:A:LOGIC:THRESHOLD?` might return `:TRIGGER:A:LOGIC:THRESHOLD CH1 24.0000E-03; CH2 1.2000E+00;CH3 1.2000E+00; CH4 1.2000E+00`, indicating the threshold voltages for the channels in an A logic trigger are as follows: channel 1 = 24 mV; channel 2 = 1.2 V; channel 3 = 1.2 V; channel 4 = 1.2 V.

TRIGger:{A|B}:LOGIc:THReshold:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the A or B logic trigger threshold voltage for the channel, specified by x, which ranges from 1 through 4. This command is equivalent to selecting A or B Event Trigger Setup from the Trig menu, choosing a logic trigger type, such as State or Pattern, and setting the Input Threshold voltage for the desired channel.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOGIc:THReshold:CH<x> <NR3>
TRIGger:{A|B}:LOGIc:THReshold:CH<x>?
```

Related Commands TRIGger:{A|B}:LOGIc:INPut?

Arguments `<NR3>` specifies the threshold voltage.

Examples `TRIGGER:A:LOGICTHRESHOLD: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|B}:LOWerthreshold:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the A or B lower trigger level for TRIGger:LVLSrcpreferenceTRIGger:LVLSrcpreference SRCDependent or

SRCIndependent modes for the channel, specified by x, which ranges from 1 through 4.

Group Trigger

Syntax
```
TRIGger:{A|B}:LOWerthreshold:CH<x> {ECL|TTL|<NR3>}
TRIGger:{A|B}:LOWerthreshold:CH<x>?
```

Related Commands TRIGger:LVLSrcpreference, TRIGger:{A|B}:UPPerthreshold:CH<x>

Arguments `ECL` specifies the preset ECL high level.

`TTL` specifies the preset TTL high level.

`<NR3>` specifies the threshold voltage in user units.

Examples `TRIGGER:A:LOWERTHRESHOLD:CH2 TTL` sets the A trigger threshold voltage for Channel 2 to the TTL low level.

`TRIGGER:A:LOWERTHRESHOLD:CH3?` might return `:TRIGGER:A:LOWERTHRESHOLD:CH3 1.2000E+00`, indicating that the A trigger threshold voltage for Channel 3 is 1.2 V.

TRIGger:{A|B}:PULse? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the pulse trigger parameters.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse?
```

Related Commands TRIGger:{A|B}:EDGE?, TRIGger:{A|B}:LOGIc?

Examples `TRIGGER:A:PULSE?` might return
```
:TRIGGER:A:PULSE:CLASS GLITCH;SOURCE CH1; GLITCH:WIDTH 2.0000E-9;TRIGIF ACCEPT;POLARITY POSITIVE; POLARITY:CH1 POSITIVE;CH2 POSITIVE;CH3 POSITIVE;CH4
```

```
POSITIVE;:TRIGGER:A:PULSE:GLITCH:QUALIFY
OCCURS;:TRIGGER:A:PULSE:WINDOW:TYPE INSIDE;WHEN
OCCURS;WIDTH 2.0000E-9;:TRIGGER:A:PULSE:TRANSITION:DELTATIME
2.0000E-9;POLARITY POSITIVE;THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:TRANSITION:WHEN
SLOWERTHAN; :TRIGGER:A:PULSE:WIDTH:LOWLIMIT
2.0000E-9;HIGHLIMIT 2.0000E-9;WHEN WITHIN;POLARITY
POSITIVE;:TRIGGER:A:PULSE:TIMEOUT:POLARITY STAYSHIGH;TIME
2.0000E-9.
```

TRIGger:{A|B}:PULse:CLAss

NOTE. *Not available on some models.*

This command sets or queries the type of pulse on which to trigger. This command is equivalent to selecting the setup menu for the pulse type that you want from the Trig menu: Glitch Setup, Width Setup, Runt Setup, Timeout Setup, or Transition Setup.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:CLAss {GLItch|RUNT|WIDth|
TRANsition|TIMEOut|WINdow}
TRIGger:{A|B}:PULse:CLAss?
```

Related Commands TRIGger:{A|B}:PULse:GLItch?, TRIGger:{A|B}:PULse:RUNT?, TRIGger:{A|B}:PULse:WIDth?, TRIGger:{A|B}:PULse:TIMEOut?, TRIGger:{A|B}:PULse:TRANsition?, TRIGger:{A|B}:TYPe

Arguments

`GLItch` triggers when a pulse is found that is of the specified polarity and width. These are set with the commands TRIGger:{A|B}:PULse:GLItch:POLarity and TRIGger:{A|B}:PULse:GLItch:WIDth.

`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 TRIGger:{A|B}:PULse:RUNT:THReshold:LOW and TRIGger:{A|B}:PULse:RUNT:THReshold:HIGH commands.

`WIDth` triggers when a pulse is found that has the specified polarity and is either inside or outside the limits as specified by TRIGger:{A|B}:PULse:WIDth:LOWLimit and TRIGger:{A|B}:PULse:WIDth:HIGHLimit. The polarity is selected using the TRIGger:{A|B}:PULse:WIDth:POLarity command.

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.

TIMEOut triggers when the pulse train stops in the selected state for longer than the specified time.

WINdow triggers when a pulse is found that meets the conditions set by the A Event window trigger type, specified by the following commands: TRIGger:{A|B}:PULse:WINdow:THReshold:HIGH, TRIGger:{A|B}:PULse:WINdow:THReshold:LOW, TRIGger:{A|B}:PULse:WINdow:THReshold:BOTh, , and TRIGger:{A|B}:PULse:WINdow:WIDTH

Examples

TRIGGER:A:PULSE:CLASS WIDTH specifies a width pulse for the A trigger.

TRIGGER:A:PULSE:CLASS? might return :TRIGGER:A:PULSE:CLASS GLITCH, indicating that a pulse was found that is of the specified polarity and width.

TRIGger:{A|B}:PULse:GLItch? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the current glitch pulse trigger parameters. This command query is equivalent to selecting Glitch Setup from the Trig menu and viewing the current glitch trigger settings.

Group

Trigger

Syntax

TRIGger:{A|B}:PULse:GLItch?

Related Commands

TRIGger:{A|B}:PULse:GLItch:POLarity, TRIGger:{A|B}:PULse:GLItch:TRIGIF, TRIGger:{A|B}:PULse:GLItch:WIDth

Examples

TRIGGER:A:PULSE:GLITCH? might return :TRIGGER:A:PULSE:GLITCH:WIDTH 2.0000E-09; FILTER ACCEPT;POLARITY POSITIVE.

TRIGger:{A|B}:PULse:GLItch:LOWPASSfilter

NOTE. *Not available on some models.*

This command turns on or turns off the lowpass filter feature for Glitch trigger. This allows triggering in the presence of highfrequency signal edges.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:GLItch:LOWPASSfilter {ON|OFF}
TRIGger:{A|B}:PULse:GLItch:LOWPASSfilter?
```

Arguments

`ON` enables lowpass filter feature.

`OFF` disables lowpass filter feature.

Examples

`TRIGGER:A:PULSE:GLITCH:LOWPASSFILTER ON` turns on the lowpass filter feature for Glitch trigger.

`TRIGGER:A:PULSE:GLITCH:LOWPASSFILTER?` might return `:TRIGGER:A:PULSE:GLITCH:LOWPASSFILTER OFF`, indicating that the lowpass filter feature is disabled for Glitch trigger.

TRIGger:{A|B}:PULse:GLItch:POLarity

NOTE. *Not available on some models.*

This command sets or queries the polarity for A or B pulse glitch trigger. This command is equivalent to selecting Glitch Setup from the Trig menu and then choosing the desired Polarity.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:GLItch:POLarity
{POSITIVe|NEGAtive|EITher}
TRIGger:{A|B}:PULse:GLItch:POLarity?
```

Related Commands TRIGger:{A|B}:LEVel, TRIGger:{A|B}:PULse:GLItch:WIDth

Arguments

`POSITIVe` specifies that the instrument will only trigger when the polarity of the glitch is positive.

`NEGAtive` specifies that the instrument will only trigger when the polarity of the glitch is negative.

EITher specifies that the instrument will trigger when the polarity of the glitch is either positive or negative.

Examples

TRIGGER:A:PULSE:GLITCH:POLARITY EITHER specifies that the polarity of the glitch can be either positive or negative for the A trigger to occur.

TRIGGER:A:PULSE:GLITCH:POLARITY? might return :TRIGGER:A:PULSE:GLITCH:POLARITY POSITIVE, indicating that the polarity of the glitch must be positive for the trigger to occur.

TRIGger:{A|B}:PULse:GLItch:POLarity:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the polarity for A or B pulse glitch trigger for the channel.

This command is equivalent to selecting Glitch Setup from the Trig menu and then choosing the desired Polarity.

Group

Trigger

Syntax

```
TRIGger:{A|B}:PULse:GLItch:POLarity:CH<x>
{EITher|NEGAtive|POSITIVe}
TRIGger:{A|B}:PULse:GLItch:POLarity:CH<x>?
```

Related Commands

TRIGger:{A|B}:LEVel, TRIGger:{A|B}:PULse:GLItch:WIDth

Arguments

EITher specifies that the instrument will trigger when the polarity of the glitch is either positive or negative.

NEGAtive specifies that the instrument will only trigger when the polarity of the glitch is negative.

POSITIVe specifies that the instrument will only trigger when the polarity of the glitch is positive.

Examples

TRIGGER:A:PULSE:GLITCH:POLARITY:CH2 EITHER specifies that the polarity of the glitch can be either positive or negative for the Channel 2 trigger to occur.

`TRIGGER:A:PULSE:GLITCH:POLARITY:CH2?` might return `:TRIGGER:A:PULSE:GLITCH:POLARITY:CH2 POSITIVE`, indicating that the polarity of the glitch must be positive for the Channel 2 trigger to occur.

TRIGger:{A|B}:PULse:GLItch:QUAlify

NOTE. *Not available on some models.*

This command sets or queries the Glitch Trigger qualification. This is equivalent to selecting Glitch Setup from the Trig menu and selecting Occurs, Logic, or Bus in the Trigger If Glitch drop-down list box.

NOTE. *If you use the LOGIc argument, you must specify the channel logic conditions.*

BUS is same as OccursAnd in the user interface.

NOTE. *In Trigger B pulse glitch, the OccursAnd option is not available in the Trigger If Glitch drop-down list box.*

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:GLItch:QUAlify {OCCurs|LOGIc|BUS}
TRIGger:{A|B}:PULse:GLItch:QUAlify?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments `OCCurs` specifies a trigger if any detectable event occurs.

`LOGIc` specifies a trigger if the individual channel qualifications meet the logic patterns and thresholds set by the TRIGger:{A|B}:LOGIc:INPut:CH<x> and TRIGger:{A|B}:LOGIc:THReshold:CH<x> commands.

`BUS` specifies a trigger if the bus input qualifications meet the pattern set by the TRIGger:QUALification:BUS:VALue command.

Examples `TRIGGER:A:PULSE:GLITCH:QUALIFY OCCURS` This command sets the trigger qualifier to OCCURS, which initiates a trigger if the instrument detects any glitch trigger qualifications.

TRIGGER:A:PULSE:GLITCH:QUALIFY? might return :TRIGGER:A:PULSE:GLITCH:QUALIFY LOGIC, indicating that a trigger will occur if the instrument detects LOGIC glitch qualifications.

TRIGger:{A|B}:PULse:GLItch:TRIGIF

NOTE. *Not available on some models.*

This command sets or queries the acceptance or rejection of the glitch pulse trigger, based on width. This command is equivalent to selecting Glitch Setup from the Trig menu and choosing the desired Trig if Width setting.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:GLItch:TRIGIF {ACCept|REJect}
TRIGger:{A|B}:PULse:GLItch:TRIGIF?
```

Related Commands TRIGger:{A|B}:PULse:GLItch:WIDth

Arguments

ACCept specifies that the instrument will only trigger on pulses that are narrower than the specified width, when the trigger type is set to glitch. The width is specified using the TRIGger:{A|B}:PULse:GLItch:WIDth command.

REJect specifies that the instrument will only trigger on pulses that are wider than the specified width, when the trigger type is set to glitch. The width is specified using the TRIGger:{A|B}:PULse:GLItch:WIDth command.

Examples

TRIGGER:A:PULSE:GLITCH:TRIGIF REJECT specifies that the instrument triggers on pulses that are wider than the specified width.

TRIGGER:A:PULSE:GLITCH:TRIGIF ACCEPT? might return :TRIGGER:A:PULSE:GLITCH:TRIGIF ACCEPT, indicating that the instrument is set to trigger on pulses that are narrower than the specified width.

TRIGger:{A|B}:PULse:GLItch:WIDth

NOTE. *Not available on some models.*

This command sets or queries the width for the glitch trigger. This command is equivalent to selecting Glitch Setup from the Trig menu and then setting the desired Width.

For information about using the width value, refer to the TRIGger:{A|B}:PULse:GLItch:TRIGIF command.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:GLItch:WIDth <NR3>
TRIGger:{A|B}:PULse:GLItch:WIDth?
```

Related Commands TRIGger:{A|B}:LEVel, TRIGger:{A|B}:PULse:GLItch:TRIGIF

Arguments <NR3> argument specifies the width of the glitch in seconds.

Examples `TRIGGER:A:PULSE:GLITCH:WIDTH 15E-6` sets the width of the glitch to 15 µs.

`TRIGGER:A:PULSE:GLITCH:WIDTH?` might return `:TRIGGER:A:PULSE:GLITCH:WIDTH 2.0000E-09`, indicating that the width of the glitch is currently set at 2 ns.

TRIGger:{A|B}:PULse:RUNT? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the current runt trigger parameters. This command query is equivalent to selecting Runt Setup from the Trig menu and then viewing the current settings.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:RUNT?
```

Related Commands TRIGger:{A|B}:PULse:GLItch?, TRIGger:{A|B}:PULse:TIMEOut?, TRIGger:{A|B}:PULse:TRANsition?, TRIGger:{A|B}:PULse:WIDth?

Examples `TRIGGER:A:PULSE:RUNT?` might return
`:TRIGGER:A:PULSE:RUNT:POLARITY POSITIVE;POLARITY:CH1`

```
POSITIVE;CH2 POSITIVE;CH3 POSITIVE;CH4
POSITIVE;:TRIGGER:A:PULSE:RUNT:THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:RUNT:WHEN OCCURS;WIDTH
500.0000E-12;QUALIFY OCCURS.
```

TRIGger:{A|B}:PULse:RUNT:POLarity

NOTE. *Not available on some models.*

This command sets or queries the polarity for the A or B pulse runt trigger. This command is equivalent to selecting Runt Setup from the Trig menu and then choosing the Polarity setting.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:RUNT:POLarity {POSITIVe|NEGAtive|EITher}
TRIGger:{A|B}:PULse:RUNT:POLarity?
```

Related Commands TRIGger:{A|B}:PULse:RUNT:THReshold?

Arguments

`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.

`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.

`EITher` indicates either negative or positive polarity.

Examples

`TRIGGER:A:PULSE:RUNT:POLARITY NEGATIVE` sets the polarity of the A pulse runt trigger to negative.

`TRIGGER:A:PULSE:RUNT:POLARITY?` might return `:TRIGGER:A:PULSE:RUNT:POLARITY POSITIVE`, indicating that the polarity of the A pulse runt trigger is set to positive.

TRIGger:{A|B}:PULse:RUNT:POLarity:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the polarity for the A or B pulse runt trigger for the channel.

This command is equivalent to selecting Runt Setup from the Trig menu and then choosing the Polarity setting for the channel.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:RUNT:POLarity:CH<x>
{EITher|NEGAtive|POSITIVe}
TRIGger:{A|B}:PULse:RUNT:POLarity:CH<x>?
```

Related Commands TRIGger:{A|B}:PULse:RUNT:THReshold?

Arguments `EITher` indicates either negative or positive polarity.

`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.

`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.

Examples `TRIGGER:A:PULSE:RUNT:POLARITY:CH3 NEGATIVE` sets the polarity of the Channel 3 A pulse runt trigger to negative.

`TRIGGER:A:PULSE:RUNT:POLARITY:CH3?` might return `:TRIGGER:A:PULSE:RUNT:POLARITY:CH3 POSITIVE`, indicating that the polarity of the A pulse runt trigger is set to positive.

TRIGger:{A|B}:PULse:RUNT:QUALify

NOTE. *Not available on some models.*

This command sets or queries the Runt Trigger qualification. This is equivalent to selecting Runt Setup from the Trig menu and selecting Occurs, Logic, or Bus in the Trigger If Runt drop-down list box.

NOTE. *If you use the LOGIc argument, you must specify the channel logic conditions.*

BUS is same as OccursAnd in the user interface.

NOTE. *In Trigger B pulse runt, the OccursAnd option is not available in the Trigger If Runt drop-down list box.*

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:RUNT:QUAlify {OCCurs|LOGIc|BUS}
TRIGger:{A|B}:PULse:RUNT:QUAlify?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments `OCCurs` specifies a trigger if any detectable event occurs.

`LOGIc` specifies a trigger if the individual channel qualifications meet the logic patterns and thresholds set by the command.

`BUS` specifies a trigger if the bus input qualifications meet the pattern set by the TRIGger:QUALification:BUS:VALue command.

Examples `TRIGGER:A:PULSE:RUNT:QUALIFY OCCURS` sets the A trigger to trigger if a qualifying runt occurs.

`TRIGGER:A:PULSE:RUNT:QUALIFY?` might return `:TRIGGER:A:PULSE:RUNT:QUALIFY OCCURS`, indicating that a trigger will occur if the instrument detects any of the runt qualifications.

TRIGger:{A|B}:PULse:RUNT:THReshold? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the upper and lower thresholds for the pulse runt trigger. This command query is equivalent to selecting Runt Setup from the Trig menu and then viewing the Upper Level and Lower Level settings.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:RUNT:THReshold?
```

Related Commands TRIGger:{A|B}:PULse:RUNT:POLarity, TRIGger:{A|B}:PULse:RUNT:THReshold:BOTh, TRIGger:{A|B}:PULse:RUNT:THReshold:HIGH, TRIGger:{A|B}:PULse:RUNT:THReshold:LOW

Examples `TRIGGER:A:PULSE:RUNT:THRESHOLD?` might return `:TRIGGER:A:PULSE:THRESHOLD:HIGH 2.0000E+00; LOW 8.0000E-01`, indicating that the upper threshold is 2 V and that the lower threshold is set to 0.8 V.

TRIGger:{A|B}:PULse:RUNT:THReshold:BOTh (No Query Form)

NOTE. *Not available on some models.*

This command (no query form) sets the upper and lower switching thresholds for the pulse runt trigger. This command is equivalent to selecting Runt Setup from the Trig menu and then setting the Upper Level and Lower Level voltages.

Group Trigger

Syntax `TRIGger:{A|B}:PULse:RUNT:THReshold:BOTh {TTL|ECL}`

Related Commands TRIGger:{A|B}:PULse:RUNT:THReshold?

Arguments `TTL` sets the upper and lower threshold to the nominal `TTL` voltage levels.

`ECL` sets the upper and lower threshold to the nominal `ECL` voltage levels.

Examples `TRIGGER:A:PULSE:RUNT:THRESHOLD:BOTH TTL` sets the threshold of the pulse runt trigger to the nominal TTL voltage levels.

TRIGger:{A|B}:PULse:RUNT:THReshold:HIGH

NOTE. *Not available on some models.*

This command sets or queries the upper limit for the pulse runt trigger. This command is equivalent to selecting Runt Setup from the Trig menu and setting the runt trigger Upper Level voltage.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:RUNT:THReshold:HIGH <NR3>
TRIGger:{A|B}:PULse:RUNT:THReshold:HIGH?
```

Related Commands TRIGger:{A|B}:PULse:RUNT:THReshold?

Arguments <NR3> specifies the threshold value in volts.

Examples `TRIGGER:A:PULSE:RUNT:THRESHOLD:HIGH 120E-3` sets the upper limit of the pulse runt trigger to 120 mV.

`TRIGGER:A:PULSE:RUNT:THRESHOLD:HIGH?` might return `:TRIGGER:A:PULSE:RUNT:THRESHOLD:HIGH 1.1000E+00`, indicating that the upper limit of the pulse runt trigger is currently set to 1.1 V.

TRIGger:{A|B}:PULse:RUNT:THReshold:LOW

NOTE. *Not available on some models.*

This command sets or queries the lower limit for the pulse runt trigger. This command is equivalent to selecting Runt Setup from the Trig menu and then setting the Lower Level voltage.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:RUNT:THReshold:LOW <NR3>
TRIGger:{A|B}:PULse:RUNT:THReshold:LOW?
```

Related Commands TRIGger:{A|B}:PULse:RUNT:THReshold?

Arguments <NR3> specifies the threshold value in volts.

Examples `TRIGGER:A:PULSE:RUNT:THRESHOLD:LOW 50E-3` sets the lower limit of the pulse runt trigger to 50 mV.

`TRIGGER:A:PULSE:RUNT:THRESHOLD:LOW?` might return `:TRIGGER:A:PULSE:RUNT:THRESHOLD:LOW 1.2000E-01`, indicating that the lower limit of the pulse runt trigger is set to 120 mV.

TRIGger:{A|B}:PULse:RUNT:WHEn

NOTE. *Not available on some models.*

This command sets or queries the type of pulse width the trigger checks for when it detects a runt. This is equivalent to selecting Runt Setup from the Trig menu and choosing the desired Trigger When setting from the drop-down list.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:RUNT:WHEn {OCCurs|WIDERthan}
TRIGger:{A|B}:PULse:RUNT:WHEn?
```

Related Commands TRIGger:{A|B}:PULse:RUNT:WIDth

Arguments

`OCCurs` argument specifies a trigger event if a runt of any detectable width occurs.

`WIDERthan` specifies a trigger event if a runt greater than the specified width occurs.

Examples

`TRIGGER:A:PULSE:RUNT:WHEN WIDERTHAN` sets the runt trigger to occur when the instrument detects a runt in a pulse wider than the specified width.

`TRIGGER:A:PULSE:RUNT:WHEN?` might return `:TRIGGER:A:PULSE:RUNT:WHEN OCCURS`, indicating that a runt trigger will occur if the instrument detects a runt of any detectable width.

TRIGger:{A|B}:PULse:RUNT:WIDth

NOTE. *Not available on some models.*

This command sets or queries the minimum width for an Pulse Runt trigger. This command is equivalent to selecting Runt Setup from the Trig menu and then setting the Width.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:RUNT:WIDth <NR3>
TRIGger:{A|B}:PULse:RUNT:WIDth?
```

Related Commands TRIGger:{A|B}:PULse:RUNT:WHEn

Arguments <NR3> specifies the minimum width in seconds.

Examples TRIGGER:A:PULSE:RUNT:WIDTH 15E-6 sets the minimum width of the pulse runt trigger to 15 µs.

TRIGGER:A:PULSE: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|B}:PULse:SOUrce

NOTE. *Not available on some models.*

This command sets or queries the source for the pulse trigger. This source parameter applies to all classes of pulse triggers. This command is equivalent to selecting Event Trigger Setup from the Trig menu, selecting the pulse type (Glitch, Width, Runt, Timeout, or Transition), and then choosing the desired channel from the Source pull-down list.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:SOUrce {CH<x>|D<x>}
TRIGger:{A|B}:PULse:SOUrce?
```

Related Commands TRIGger:{A|B}:EDGE:SOUrce

Arguments CH<x> specifies one of the input channels, which range from 1 through 4.

D<x> specifies one of the digital inputs, which range from 0 through 15.

Examples TRIGGER:A:PULSE:SOURCE CH4 sets Channel 4 as the source for the A pulse trigger.

TRIGGER:A:PULSE:SOURCE? might return :TRIGGER:A:PULSE:SOURCE CH2, indicating that the Channel 2 is the source for the A pulse trigger.

TRIGger:{A|B}:PULse:TIMEOut? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the polarity and time-out duration for the pulse timeout trigger. This command is equivalent to selecting Timeout Setup from the Trig menu and viewing the polarity in the Trigger When box and the Timer setting.

Group Trigger

Syntax `TRIGger:{A|B}:PULse:TIMEOut?`

Related Commands TRIGger:{A|B}:PULse:TIMEOut:POLarity, TRIGger:{A|B}:PULse:TIMEOut:TIMe

Examples `TRIGGER:A:PULSE:TIMEOUT?` might return `:TRIGGER:A:PULSE:TIMEOUT:POLARITY STAYSHIGH;TIME 2.0000E-9`, indicating that the polarity setting for the A pulse trigger is STAYSHIGH (positive) and the timeout duration is 2 ns.

TRIGger:{A|B}:PULse:TIMEOut:LOWPASSfilter

NOTE. *Not available on some models.*

This command turns on or turns off the lowpass filter feature for Timeout trigger. This allows triggering in the presence of high frequency signal edges.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:TIMEOut:LOWPASSfilter {ON|OFF}
TRIGger:{A|B}:PULse:TIMEOut:LOWPASSfilter?
```

Arguments `ON` enables lowpass filter feature.

`OFF` disables lowpass filter feature.

Examples `TRIGGER:A:PULSE:TIMEOUT:LOWPASSFILTER ON` turns on the lowpass filter feature for Timeout trigger.

TRIGGER:A:PULSE:TIMEOUT:LOWPASSFILTER? might return :TRIGGER:A:PULSE:TIMEOUT:LOWPASSFILTER OFF, indicating that the lowpass filter feature is disabled for Timeout trigger.

TRIGger:{A|B}:PULse:TIMEOut:POLarity

NOTE. *Not available on some models.*

This command sets or queries the polarity for the pulse timeout trigger. This command is equivalent to selecting Timeout Setup from the Trig menu and setting the desired polarity in the Trigger When box.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:TIMEOut:POLarity
{STAYSHigh|STAYSLow|EITher}
TRIGger:{A|B}:PULse:TIMEOut:POLarity?
```

Related Commands TRIGger:{A|B}:PULse:TIMEOut?

Arguments

STAYSHigh indicates that a pulse edge must stay high (positive) for the required time period to permit timeout triggering to occur. This is the default polarity.

STAYSLow indicates that a pulse edge must stay low (negative) for the required time period to permit timeout triggering to occur.

EITher indicates that the polarity of the timeout trigger can stay either high or low (positive or negative) for the required time period to permit time out triggering to occur.

Examples

TRIGGER:A:PULSE:TIMEOUT:POLARITY STAYSHIGH sets the polarity of the A timeout trigger to positive.

TRIGGER:A:PULSE:TIMEOUT:POLARITY? might return :TRIGGER:A:PULSE:TIMEOUT:POLARITY EITHER, indicating that the polarity of the A timeout trigger can be either positive or negative.

TRIGger:{A|B}:PULse:TIMEOut:POLarity:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the polarity for the A or B pulse timeout trigger for the channel.

This command is equivalent to selecting Transition Setup from the Trig menu and then setting the desired Polarity for the channel.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:TIMEOut:POLarity:CH<x>
{STAYSHigh|STAYSLow|EITher}
TRIGger:{A|B}:PULse:TIMEOut:POLarity:CH<x>?
```

Arguments

`STAYSHigh` indicates that a pulse edge must stay high (positive) for the required time period to permit time out triggering to occur. This is the default polarity.

`STAYSLow` indicates that a pulse edge must stay low (negative) for the required time period to permit time out triggering to occur.

`EITher` indicates that the polarity of the time out trigger can stay either high or low (positive or negative) for the required time period to permit time out triggering to occur.

Examples

`TRIGGER:A:PULSE:TIMEOUT:POLARITY:CH1 STAYSHIGH` sets the polarity of the A pulse time out trigger to positive for Channel 1.

`TRIGGER:A:PULSE:TIMEOUT:POLARITY:CH1?` might return `:TRIGGER:A:PULSE:TIMEOUT:POLARITY:CH1 EITHER`, indicating that the polarity of the A pulse time out trigger for Channel 1 can be either positive or negative.

TRIGger:{A|B}:PULse:TIMEOut:QUAlify

NOTE. *Not available on some models.*

This command sets or queries the Timeout Trigger qualification.

This is equivalent to selecting Timeout Setup from the Trig menu and selecting Occurs, Logic, or Bus in the Trigger If Timeout drop-down list box.

NOTE. *If you use the LOGIc argument, you must specify the channel logic conditions.*

BUS is same as OccursAnd in the user interface.

NOTE. *In Trigger B pulse timeout, the OccursAnd option is not available in the Trigger If Timeout drop-down list box.*

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:TIMEOut:QUAlify {OCCurs|LOGIc|BUS}
TRIGger:{A|B}:PULse:TIMEOut:QUAlify?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments `OCCurs` specifies a trigger if any detectable event occurs.

`LOGIc` specifies a trigger if the individual channel qualifications meet the logic patterns and thresholds set by the TRIGger:{A|B}:LOGIc:INPut:CH<x> and TRIGger:{A|B}:LOGIc:THReshold:CH<x> commands.

`BUS` specifies a trigger if the bus input qualifications meet the pattern set by the TRIGger:QUALification:BUS:VALue command.

Examples `TRIGGER:A:PULSE:TIMEOUT:QUALIFY OCCURS` sets the trigger qualifier to OCCURS, which initiates a trigger if the instrument detects any timeout trigger qualifications.

`TRIGGER:A:PULSE:TIMEOUT:QUALIFY?` might return `:TRIGGER:A:PULSE:TIMEOUT:QUALIFY OCCURS`, indicating that a trigger will occur if the instrument detects any of the timeout qualifications.

TRIGger:{A|B}:PULse:TIMEOut:TIMe

NOTE. *Not available on some models.*

This command sets or queries the pulse timeout trigger time (measured in seconds). This command is equivalent to selecting Timeout Setup from the Trig menu and setting a value for Timer.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:TIMEOut:TIMe <NR3>
TRIGger:{A|B}:PULse:TIMEOut:TIMe?
```

Related Commands TRIGger:{A|B}:PULse:TIMEOut?

Arguments <NR3> argument specifies the timeout period in seconds.

Examples TRIGGER:A:PULSE:TIMEOUT:TIME 3.134E-6 sets the timeout to 3.134 µs.

TRIGGER:A:;PULSE:TIMEOUT:TIME? might return :TRIGGER:A:PULSE:TIMEOUT:TIME 2.0000E-9, indicating that the timeout is currently set to 2 ns.

TRIGger:{A|B}:PULse:TRANsition? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns delta time, polarity, and both upper and lower threshold limits for the transition time trigger. This command is equivalent to selecting Transition Setup from the Trig menu and then viewing the current transition settings.

Group Trigger

Syntax TRIGger:{A|B}:PULse:TRANsition?

Related Commands TRIGger:{A|B}:TYPe, TRIGger:{A|B}:PULse:CLAss, TRIGger:{A|B}:PULse:GLItch?, TRIGger:{A|B}:PULse:RUNT?, TRIGger:{A|B}:PULse:TIMEOut?, TRIGger:{A|B}:PULse:WIDth?

Examples TRIGGER:A:PULSE:TRANSITION? might return :TRIGGER:A:PULSE:TRANSITION:DELTATIME 2.0000E-9;POLARITY POSITIVE;THRESHOLD:HIGH 1.2000;LOW 800.0000E-3;:TRIGGER:A:PULSE:TRANSITION:WHEN SLOWERTHAN indicating the current transition time trigger settings.

TRIGger:{A|B}:PULse:TRANsition:DELTATime

NOTE. *Not available on some models.*

This command sets or queries the delta time used in calculating the transition value for the transition trigger. This is equivalent to selecting Transition Setup from the Trig menu and setting the Time.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:TRANsition:DELTATime <NR3>
TRIGger:{A|B}:PULse:TRANsition:DELTATime?
```

Related Commands TRIGger:{A|B}:PULse:TRANsition:POLarity, TRIGger:{A|B}:PULse:TRANsition:THReshold?

Arguments <NR3> specifies the delta time in seconds.

Examples `TRIGGER:A:PULSE:TRANSITION:DELTATIME 15E-6` sets the delta time of the transition trigger to 15 µs.

`TRIGGER:A:PULSE:TRANSITION:DELTATIME?` might return `:TRIGGER:A:PULSE:TRANSITION:DELTATIME 2.0000E-09`, indicating that the delta time of the transition trigger is set to 2 ns.

TRIGger:{A|B}:PULse:TRANsition:POLarity

NOTE. *Not available on some models.*

This command sets or queries the polarity for the transition trigger. This command is equivalent to selecting Transition Setup from the Trig menu and choosing from the Polarity drop-down list.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:TRANsition:POLarity
{POSITIVe|NEGAtive|EITher}
TRIGger:{A|B}:PULse:TRANsition:POLarity?
```

Related Commands TRIGger:{A|B}:PULse:TRANsition:DELTATime

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:PULSE:TRANSITION:POLARITY NEGATIVE sets the transition polarity to negative.

TRIGGER:A:PULSE:TRANSITION:POLARITY? might return :TRIGGER:A:PULSE:TRANSITION:POLARITY EITHER, indicating that the polarity can be either positive or negative.

TRIGger:{A|B}:PULse:TRANsition:POLarity:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the polarity for the A or B pulse transition trigger for the channel.

This command is equivalent to selecting Transition Setup from the Trig menu and then choosing from the Polarity pull-down list for the channel.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:TRANsition:POLarity:CH<x>
{EITher|NEGAtive|POSITIVe}
TRIGger:{A|B}:PULse:TRANsition:POLarity:CH<x>?
```

Related Commands TRIGger:{A|B}:PULse:TRANsition:DELTATime

Arguments EITher indicates either positive or negative polarity.

NEGAtive indicates that a pulse edge must traverse from the upper (most positive) to lower (most negative) level for transition triggering to occur.

POSITIVe indicates that a pulse edge must traverse from the lower (most negative) to higher (post positive) level for transition triggering to occur.

Examples TRIGGER:A:PULSE:TRANSITION:POLARITY NEGATIVE sets the trigger A transition polarity to negative.

TRIGGER:A:PULSE:TRANSITION:POLARITY? might return :TRIGGER:A:PULSE:TRANSITION:POLARITY EITHER, indicating that the transition can be either positive or negative for trigger A.

TRIGger:{A|B}:PULse:TRANsition:QUAlify

NOTE. *Not available on some models.*

This command sets or queries the Transition Time Trigger qualification.

This is equivalent to selecting Transition Setup from the Trig menu and selecting Occurs, Logic, or Bus in the Trigger If Transition drop-down list box.

NOTE. *If you use the LOGIc argument, you must specify the channel logic conditions.*

BUS is same as OccursAnd in the user interface.

NOTE. *In Trigger B pulse transition, the OccursAnd option is not available in the Trigger If Transition drop-down list box.*

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:TRANsition:QUAlify {OCCurs|LOGIc|BUS}
TRIGger:{A|B}:PULse:TRANsition:QUAlify?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments

OCCurs specifies a trigger if any violations occur.

LOGIc specifies a trigger if the individual channel qualifications meet the logic patterns and thresholds set by the TRIGger:{A|B}:LOGIc:INPut:CH<x> and TRIGger:{A|B}:LOGIc:THReshold:CH<x> commands.

BUS specifies a trigger if the bus input qualifications meet the pattern set by the TRIGger:QUALification:BUS:VALue command.

Examples TRIGGER:A:PULSE:TRANSITION:QUALIFY OCCURS sets the trigger qualifier to OCCurs, which initiates an A trigger if the instrument detects any transition violations.

TRIGGER:A:PULSE:TRANSITION:QUALIFY? might return :TRIGGER:A:PULSE:TRANSITION:QUALIFY OCCURS, indicating that an A trigger will occur if the instrument detects any transition violations.

TRIGger:{A|B}:PULse:TRANsition:THReshold? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the upper and lower threshold limits for the transition time trigger. This command is equivalent to selecting Transition Setup from the Trig menu and viewing the Upper Level and Lower Level voltage settings.

Group Trigger

Syntax TRIGger:{A|B}:PULse:TRANsition:THReshold?

Related Commands TRIGger:{A|B}:PULse:TRANsition:DELTATime, TRIGger:{A|B}:PULse:TRANsition:POLarity

Examples TRIGGER:A:PULSE:TRANSITION:THRESHOLD? might return :TRIGGER:A:PULSE:TRANSITION:THRESHOLD:HIGH 1.2000;LOW 800.0000E-3, indicating that the upper threshold limit to 1.2 V and lower threshold limit for the transition time trigger to 0.8 V.

TRIGger:{A|B}:PULse:TRANsition:THReshold:BOTh (No Query Form)

NOTE. *Not available on some models.*

This command (no query form) sets the upper and lower thresholds for the transition trigger. This command is equivalent to selecting Transition Setup from the Trig menu and setting the desired Upper Level and Lower Level voltages.

Group Trigger

Syntax TRIGger:{A|B}:PULse:TRANsition:THReshold:BOTh {TTL|ECL}

Related Commands TRIGger:{A|B}:PULse:TRANsition:THReshold:HIGH, TRIGger:{A|B}:PULse:TRANsition:THReshold:LOW

Arguments `TTL` sets the upper and lower threshold to the nominal `TTL` voltage levels.

`ECL` sets the upper and the lower threshold to the nominal `ECL` voltage levels.

Examples `TRIGGER:A:PULSE:TRANSITION:THRESHOLD:BOTH TTL` sets the thresholds of the transition trigger to the nominal TTL voltage levels.

TRIGger:{A|B}:PULse:TRANsition:THReshold:HIGH

NOTE. *Not available on some models.*

This command sets or queries the upper (most positive) transition trigger threshold. This command is equivalent to selecting Transition Setup from the Trig menu and then setting the desired Upper Level voltage.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:TRANsition:THReshold:HIGH <NR3>
TRIGger:{A|B}:PULse:TRANsition:THReshold:HIGH?
```

Related Commands TRIGger:{A|B}:PULse:TRANsition:THReshold:LOW

Arguments `<NR3>` specifies the upper threshold in volts.

Examples `TRIGGER:A:PULSE:TRANSITION:THRESHOLD:HIGH 120E-3` sets the upper limit of the pulse transition trigger to 120 mV.

`TRIGGER:A:PULSE:TRANSITION:THRESHOLD:HIGH?` might return `:TRIGGER:A:PULSE:TRANSITION:THRESHOLD:HIGH 2.0000E+00`, indicating that the upper limit of the pulse transition trigger is 2 V.

TRIGger:{A|B}:PULse:TRANsition:THReshold:LOW

NOTE. *Not available on some models.*

This command sets or queries the lower (most negative) transition trigger threshold. This command is equivalent to selecting Transition Setup from the Trig menu and setting the desired Lower Level voltage.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:TRANsition:THReshold:LOW <NR3>
TRIGger:{A|B}:PULse:TRANsition:THReshold:LOW?
```

Related Commands TRIGger:{A|B}:PULse:TRANsition:THReshold:HIGH

Arguments <NR3> specifies the lower threshold in volts.

Examples TRIGGER:A:PULSE:TRANSITION:THRESHOLD:LOW 20E-3 sets the lower limit of the transition trigger to 20 mV.

TRIGGER:A:PULSE:TRANSITION:THRESHOLD:LOW? might return :TRIGGER:A:PULSE:TRANSITION:THRESHOLD:LOW 50.0000E-03, indicating that the lower limit of the transition trigger is 50 mV.

TRIGger:{A|B}:PULse:TRANsition:WHEn

NOTE. *Not available on some models.*

This command sets or queries whether to check for a transitioning signal that is faster or slower than the specified delta time. This is equivalent to selecting Transition Setup from the Trig menu and choosing the Trigger When Transition Time setting.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:TRANsition:WHEn {FASTERthan|SLOWERthan}
TRIGger:{A|B}:PULse:TRANsition:WHEn?
```

Related Commands TRIGger:{A|B}:PULse:TRANsition:DELTATime, TRIGger:{A|B}:PULse:TRANsition:POLarity, TRIGger:{A|B}:PULse:TRANsition:THReshold?

Arguments FASTERthan sets the trigger to occur when the transitioning signal is faster than the set volts/second rate.

SLOWERthan sets the trigger to occur when the transitioning signal is slower than the set volts/second rate.

Examples

TRIGGER:A:PULSE:TRANSITION:WHEN SLOWERTHAN sets the trigger to occur when the transitioning signal is slower than the set volts/second rate.

TRIGGER:A:PULSE:TRANSITION:WHEN? might return :TRIGGER:A:PULSE:TRANSITION:WHEN FASTERTHAN, indicating that the transition triggers when the transitioning signal is faster than the set volts/second rate.

TRIGger:{A|B}:PULse:WIDth? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the width parameters for the pulse width trigger. This command is equivalent to selecting Width Setup from the Trig menu and then viewing the current pulse width trigger Lower Limit, Upper Limit, Trig When and Polarity settings.

Group

Trigger

Syntax

TRIGger:{A|B}:PULse:WIDth?

Related Commands

TRIGger:{A|B}:PULse:WIDth:HIGHLimit, TRIGger:{A|B}:PULse:WIDth:LOWLimit, TRIGger:{A|B}:PULse:WIDth:POLarity, TRIGger:{A|B}:PULse:WIDth:WHEn

Examples

TRIGGER:A:PULSE:WIDTH? might return :TRIGGER:A:PULSE:WIDTH:LOWLIMIT 2.0000E-9; HIGHLIMIT 2.0000E-9;WHEN WITHIN; POLARITY POSITIVE as the current A width trigger parameters.

TRIGger:{A|B}:PULse:WIDth:HIGHLimit

NOTE. *Not available on some models.*

This command sets or queries the upper limit for the width trigger. This command is equivalent to selecting Width Setup from the Trig menu and setting the Upper Limit.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WIDth:HIGHLimit <NR3>
TRIGger:{A|B}:PULse:WIDth:HIGHLimit?
```

Related Commands TRIGger:{A|B}:PULse:WIDth?

Arguments <NR3> specifies the width trigger upper limit in seconds.

Examples `TRIGGER:A:PULSE:WIDTH:HIGHLIMIT 5.0E-6` sets the upper limit of the width trigger to 5 µs.

`TRIGGER:A:PULSE:WIDTH:HIGHLIMIT?` might return `:TRIGGER:A:PULSE:WIDTH:HIGHLIMIT 2.0000E-9`, indicating that the upper limit of the width trigger is set to 2 ns.

TRIGger:{A|B}:PULse:WIDth:LOWLimit

NOTE. *Not available on some models.*

This command sets or queries the lower limit for the width trigger. This command is equivalent to selecting Width Setup from the Trig menu and setting the pulse Lower Limit.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WIDth:LOWLimit <NR3>
TRIGger:{A|B}:PULse:WIDth:LOWLimit?
```

Related Commands TRIGger:{A|B}:PULse:WIDth?

Arguments <NR3> specifies the A pulse width trigger lower limit in seconds.

Examples `TRIGGER:A:PULSE:WIDTH:LOWLIMIT 2.0E-6` sets the lower limit of the width trigger to 2 µs.

`TRIGGER:A:PULSE:WIDTH:LOWLIMIT?` might return `:TRIGGER:A:PULSE:WIDTH:LOWLIMIT 1.0000E-9`, indicating that the lower limit of the width trigger is set to 1 ns.

TRIGger:{A|B}:PULse:WIDth:LOWPASSfilter

NOTE. *Not available on some models.*

This command turns on or turn off the lowpass filter feature for pulse width trigger. This allows triggering in the presence of high frequency signal edges.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WIDth:LOWPASSfilter {ON|OFF}
TRIGger:{A|B}:PULse:WIDth:LOWPASSfilter?
```

Arguments `ON` enables lowpass filter feature.

`OFF` disables lowpass filter feature.

Examples `TRIGGER:A:PULSE:WIDTH:LOWPASSFILTER ON` turns on the lowpass filter feature for pulse width trigger.

`TRIGGER:A:PULSE:WIDTH:LOWPASSFILTER?` might return `:TRIGGER:A:PULSE:WIDTH:LOWPASSFILTER OFF`, indicating that the lowpass filter feature is disabled for pulse width trigger.

TRIGger:{A|B}:PULse:WIDth:POLarity

NOTE. *Not available on some models.*

This command sets or queries the polarity for the width trigger. This command is equivalent to selecting Width Setup from the Trig menu and selecting the Polarity.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:WIDth:POLarity {NEGAtive|POSITIVe}
TRIGger:{A|B}:PULse:WIDth:POLarity?
```

Related Commands

TRIGger:{A|B}:PULse:WIDth?

Arguments

`NEGAtive` specifies a negative pulse.

`POSITIVe` specifies a positive pulse.

Examples

`TRIGGER:A:PULSE:WIDTH:POLARITY NEGATIVE` sets the pulse polarity to negative.

`TRIGGER:A:PULSE:WIDTH:POLARITY?` might return `:TRIGGER:A:PULSE:WIDTH:POLARITY POSITIVE`, indicating that the pulse polarity is set to positive.

TRIGger:{A|B}:PULse:WIDth:POLarity:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the polarity for the A or B pulse width trigger for the channel.

This command is equivalent to selecting Width Setup from the Trig menu and then selecting the pulse width trigger Polarity for the channel.

Group

Trigger

Syntax

```
TRIGger:{A|B}:PULse:WIDth:POLarity:CH<x> {NEGAtive|POSITIVe}
TRIGger:{A|B}:PULse:WIDth:POLarity:CH<x>?
```

Related Commands

TRIGger:{A|B}:PULse:WIDth?

Arguments

`NEGAtive` specifies a negative pulse.

`POSITIVe` specifies a positive pulse.

Examples

`TRIGGER:A:PULSE:WIDTH:POLARITY:CH4 NEGATIVE` sets the pulse polarity to negative for Channel 4.

`TRIGGER:A:PULSE:WIDTH:POLARITY:CH4?` might return `:TRIGGER:A:PULSE:WIDTH:POLARITY:CH4 POSITIVE`, indicating that the pulse polarity is set to positive for Channel 4.

TRIGger:{A|B}:PULse:WIDth:QUALify

NOTE. *Not available on some models.*

This command sets or queries the Width Trigger qualification.

This is equivalent to selecting Width Setup from the Trig menu and selecting Occurs, Logic, or Bus in the Trigger If Width drop-down list box.

NOTE. *If you use the LOGIc argument, you must specify the channel logic conditions.*

BUS is same as OccursAnd in the user interface.

NOTE. *In Trigger B pulse width, the OccursAnd option is not available in the Trigger If Width drop-down list box.*

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:WIDth:QUAlify {OCCurs|LOGIc|BUS}
TRIGger:{A|B}:PULse:WIDth:QUAlify?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments

`OCCURS` specifies a trigger if any detectable event occurs.

`LOGIC` specifies a trigger if the individual channel qualifications meet the logic patterns and thresholds set by the TRIGger:{A|B}:LOGIc:INPut:CH<x> and TRIGger:{A|B}:LOGIc:THReshold:CH<x> commands.

`BUS` specifies a trigger if the bus input qualifications meet the pattern set by the TRIGger:QUALification:BUS:VALue command.

Examples `TRIGGER:A:PULSE:WIDTH:QUALIFY OCCURS` sets the A trigger qualifier to OCCURS, which initiates a trigger if the instrument detects any width trigger qualifications.

TRIGGER:A:PULSE:WIDTH:QUALIFY? might return :TRIGGER:A:PULSE:WIDTH:QUALIFY OCCURS, indicating that an A trigger will occur if the instrument detects any of the width qualifications.

TRIGger:{A|B}:PULse:WIDth:WHEn

NOTE. *Not available on some models.*

This command sets or queries whether to trigger on a pulse width that falls outside (or within) the specified range of limits. You can define or query trigger pulse width upper and lower limits using the TRIGger:A:PULse:WIDth:HIGHLimit and TRIGger:A:PULse:WIDth:LOWLimit commands.

This command is equivalent to selecting Width Setup from the Trig menu and then choosing from the Trig When drop-down list.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:WIDth:WHEn {OUTside|WIThin}
TRIGger:{A|B}:PULse:WIDth:WHEn?
```

Related Commands TRIGger:{A|B}:PULse:WIDth:HIGHLimit, TRIGger:{A|B}:PULse:WIDth:LOWLimit

Arguments

OUTside argument causes a trigger event the duration of the pulse is greater than the high limit or less than the low limit specified. The high and low limits are specified with the TRIGger:A:PULse:WIDth:HIGHLimit and TRIGger:A:PULse:WIDth:LOWLimit commands respectively.

WIThin argument causes a trigger event when the duration of the pulse is within the high and low limits. The high and low limits are specified with the TRIGger:A:PULse:WIDth:HIGHLimit and TRIGger:A:PULse:WIDth:LOWLimit command respectively.

Examples

TRIGGER:A:PULSE:WIDTH:WHEN WITHIN specifies that the duration of the A pulse will fall within defined high and low limits.

TRIGGER:A:PULSE:WIDTH:WHEN? might return :TRIGGER:A:PULSE:WIDTH:WHEN OUTSIDE, indicating the conditions for generating a width trigger.

TRIGger:{A|B}:PULse:WINdow? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the current window trigger parameters.

Group Trigger

Syntax `TRIGger:{A|B}:PULse:WINdow?`

Examples `TRIGGER:A:PULSE:WINDOW?` might return
`:TRIGGER:A:PULSE:WINDOW:TYPE INSIDE; WHEN OCCURS;WIDTH 2.0000e-9; LOGIC:INPUT:CH1 HIGH;CH2 X; CH3 X; CH4 LOW; :TRIGGER:A:PULSE:WINDOW:LOGIC:THRESHOLD:CH1 1.2000;CH2 1.2000;CH3 1.2000;CH4 1.2000; :TRIGGER:A:PULSE:WINDOW:THRESHOLD:HIGH 1.2000;LOW 800.0000E-3`.

TRIGger:{A|B}:PULse:WINdow:EVENT

NOTE. *Not available on some models.*

This command sets or queries the window trigger event. This command is equivalent to selecting Window Setup from the Trig menu and selecting from the Window Event box.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WINdow:EVENT
{OUTSIDEGreater|INSIDEGreater|ENTERSWindow|EXITSWindow}
TRIGger:{A|B}:PULse:WINdow:EVENT?
```

Arguments `OUTSIDEGreater` specifies a trigger event when the signal leaves the window defined by the threshold levels for the time specified by Width.

`INSIDEGreater` specifies a trigger event when the signal enters the window defined by the threshold levels for the time specified by Width.

`ENTERSWindow` specifies a trigger event when the signal enters the window defined by the threshold levels.

EXITSWindow specifies a trigger event when the signal leaves the window defined by the threshold levels.

Examples

TRIGGER:A:PULSE:WINDOW:WHEN OCCURS sets the widow violation qualifier to OCCURS, which causes a trigger to occur when the instrument detects any window violation.

TRIGGER:A:PULSE:WINDOW:WHEN? might return :TRIGGER:A:PULSE:WINdow:WHEN OCCURS, indicating that a trigger will occur if the instrument detects any window violation.

TRIGger:{A|B}:PULse:WINdow:POLarity

NOTE. *Not available on some models.*

This command sets or queries the pulse trigger window polarity of the selected trigger Source. The instrument triggers when the signal exits the window. The command is available only when the option Inside > t is selected in the Window Event drop-dowm list and not available for the rest of the window events. The logic selection is available only when the polarity is set to Either.

Group

Trigger

Syntax

```
TRIGger:{A|B}:PULse:WINdow:POLarity
{EITher|NEGAtive|POSITIVe}
TRIGger:{A|B}:PULse:WINdow:POLarity?
```

Related Commands

TRIGger:{A|B}:PULse:WINdow:EVENT

Arguments

EITher specifies positive or negative polarity.

NEGAtive specifies a negative polarity.

POSITIVe specifies a positive polarity.

Examples

TRIGGER:A:PULSE:WINDOW:POLARITY POSITIVE sets the window polarity to positive.

TRIGGER:A:PULSE:WINDOW:POLARITY? might return :TRIGGER:A:PULSE:WINDOW:POLARITY EITHER, indicating that the window polarity is either positive or negative.

TRIGger:{A|B}:PULse:WINdow:POLarity:CH<x>

NOTE. *Not available on some models.*

This command sets or queries the pulse trigger window polarity of the specified channel. The oscilloscope triggers only when the signal exits the window. The command is available only when the Window Event is set to Inside > t and not available for the rest of the window events. The logic selection is available only when the polarity is set to Either. <x> is the search number or channel number.

Group Trigger

Syntax

```
TRIGger:{A|B}:PULse:WINdow:POLarity:CH<x>
{EITher|NEGAtive|POSITIVe}
TRIGger:{A|B}:PULse:WINdow:POLarity:CH<x>?
```

Arguments

`EITher` specifies positive or negative polarity.

`NEGAtive` specifies a negative polarity.

`POSITIVe` specifies a positive polarity.

Examples

`TRIGGER:A:PULSE:WINDOW:POLARITY:CH1 POSITIVE` sets the window polarity to positive.

`TRIGGER:A:PULSE:WINDOW:POLARITY:CH1?` might return `:TRIGGER:A:PULSE:WINDOW:POLARITY:CH1 EITHER`, indicating that the window polarity is either positive or negative.

TRIGger:{A|B}:PULse:WINdow:QUAlify

NOTE. *Not available on some models.*

This command sets or queries the Window Trigger qualification. This is equivalent to selecting Window Setup from the Trig menu and selecting Occurs, Logic, or Bus in the Trigger If Window drop-down list box.

NOTE. *If you use the LOGIc argument, you must specify the channel logic conditions.*

BUS is same as OccursAnd in the user interface.

NOTE. In Trigger B pulse window, the OccursAnd option is not available in the Trigger If Window drop-down list box.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WINdow:QUAlify {OCCurs|LOGIc|BUS}
TRIGger:{A|B}:PULse:WINdow:QUAlify?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments

`OCCurs` specifies a trigger if any detectable event occurs.

`LOGIc` specifies a trigger if the individual channel qualifications meet the logic patterns and thresholds set by the TRIGger:{A|B}:LOGIc:INPut:CH<x> and TRIGger:{A|B}:LOGIc:THReshold:CH<x> commands.

`BUS` specifies a trigger if the bus input qualifications meet the pattern set by the TRIGger:QUALification:BUS:VALue command.

Examples

`TRIGGER:A:PULSE:WINDOW:QUALIFY OCCURS` sets the trigger qualifier to OCCURS, which causes an A trigger to occur if the instrument detects any of the window trigger qualifications.

`TRIGGER:A:PULSE:WINDOW:QUALIFY?` might return `:TRIGGER:A:PULSE:WINDOW:QUALIFY OCCURS`, indicating that an A trigger will occur if the instrument detects any of the window qualifications.

TRIGger:{A|B}:PULse:WINdow:THReshold? (Query Only)

NOTE. Not available on some models.

This query-only command returns the upper and lower thresholds for the window trigger. This command query is equivalent to selecting Window Setup from the Trig menu and viewing the window trigger Upper Level and Lower Level settings.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WINdow:THReshold?
```

Examples `TRIGGER:A:PULSE:WINDOW:THRESHOLD?` might return `:TRIGGER:A:PULSE:WINDOW:THRESHOLD:HIGH 1.2000;LOW 800.0000E-3`, indicating that the upper threshold is 1.2 V and that the lower threshold is 800 mV.

TRIGger:{A|B}:PULse:WINdow:THReshold:BOTh (No Query Form)

NOTE. *Not available on some models.*

This command (no query form) sets the upper and lower switching thresholds for the window trigger. This command is equivalent to selecting Window Setup from the Trig menu and then setting the Upper Level and Lower Level voltages.

Group Trigger

Syntax `TRIGger:{A|B}:PULse:WINdow:THReshold:BOTh {TTL|ECL}`

Arguments `TTL` argument sets the upper and the lower threshold to the nominal `TTL` voltage levels.

`ECL` argument sets the upper and the lower threshold to the nominal `ECL` voltage levels.

Examples `TRIGGER:A:PULSE:WINDOW:THRESHOLD:BOTH TTL` sets the threshold of the window trigger to the nominal TTL voltage levels.

TRIGger:{A|B}:PULse:WINdow:THReshold:HIGH

NOTE. *Not available on some models.*

This command sets or queries the upper limit for the pulse window trigger. This command is equivalent to selecting Window Setup from the Trig menu and setting the window trigger Upper Level voltage.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WINdow:THReshold:HIGH <NR3>
TRIGger:{A|B}:PULse:WINdow:THReshold:HIGH?
```

Arguments <NR3> specifies the threshold value in volts.

Examples TRIGGER:A:PULSE:WINDOW:THRESHOLD:HIGH 2.0, sets the upper limit of the pulse window trigger to 2 V.

TRIGGER:A:PULSE:WINDOW:THRESHOLD:HIGH? might return :TRIGGER:A:PULSE:WINDOW:THRESHOLD:HIGH 1.1000E+00, indicating that the upper limit of the window trigger is set to 1.1 V.

TRIGger:{A|B}:PULse:WINdow:THReshold:LOW

NOTE. *Not available on some models.*

This command sets or queries the lower limit for the window trigger. This command is equivalent to selecting Window Setup from the Trig menu and setting the Lower Level voltage.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WINdow:THReshold:LOW <NR3>
TRIGger:{A|B}:PULse:WINdow:THReshold:LOW?
```

Arguments <NR3> specifies the threshold value in volts.

Examples TRIGGER:A:PULSE:WINDOW:THRESHOLD:LOW 0.8 sets the lower threshold of the window trigger to 0.8 V.

TRIGGER:A:PULSE:WINDOW:THRESHOLD:LOW? might return :TRIGGER:A:PULSE:WINDOW:THRESHOLD:LOW 1.2000E-01, indicating that the lower limit of the window trigger is currently set to 120 mV.

TRIGger:{A|B}:PULse:WINdow:WIDTH

NOTE. *Not available on some models.*

This command sets or queries the minimum width for a window violation. This command is equivalent to selecting Window Setup from the Trig menu, selecting Wider than in the Trigger When box, and setting the Width.

Group Trigger

Syntax
```
TRIGger:{A|B}:PULse:WINdow:WIDTH <NR3>
TRIGger:{A|B}:PULse:WINdow:WIDTH?
```

Arguments <NR3> argument specifies the minimum width in seconds.

Examples `TRIGGER:A:PULSE:WINDOW:WIDTH 15E-9` sets the minimum width of a valid window violation to 15 ns.

`TRIGGER:A:PULSE:WINDOW:WIDTH?` might return `:TRIGGER:A:PULSE:WINdow:WIDTH 2.0000E-09` indicating that the minimum width of a valid window violation is 2 ns.

TRIGger:{A|B}:READY? (Query Only)

NOTE. *Not available on some models.*

This command queries the trigger ready state and provides the immediate state from the trigger system. It is a more synchronous means of determining when the oscilloscope is ready to trigger. The TRIGger:STATE? reflects a less-frequently updated status of the trigger LEDs on the instrument front panel.

NOTE. *A delay or B trigger can become `ready´ under different circumstances. If delay by time is enabled, the B trigger becomes ready after the "trigger after time" has elapsed. If delay by events is enabled, the B trigger becomes ready after the A trigger has occurred. Also, a `ready´ state cannot be guaranteed when Reset Events are used in an A to B Event Sequence trigger.*

Group Trigger

Syntax
```
TRIGger:{A|B}:READY?
```

Related Commands TRIGger:STATE?, TRIGger:B:BY, TRIGger:B:TIMe, TRIGger:B:EVENTS:COUNt

Examples `TRIGGER:{A|B}:READY?` might return `:TRIGGER:{A|B}:READY 1`, indicating that all pretrigger information is acquired and that the instrument is ready to accept a trigger

TRIGGER:{A|B}:READY? might return :TRIGGER:{A|B}:READY 0, indicating that the instrument is not yet ready to accept a trigger

TRIGger:{A|B}:TYPe

This command sets or queries the type of A or B trigger. Logic and Pulse triggers contain classes. Logic triggers consist of State and Pattern classes; Pulse triggers consist of Glitch, Runt, Width, Transition, Timeout, and Window classes. Once you have set the trigger type, you might also need to identify the associated trigger class. For details on selecting Logic and Pulse trigger classes, see TRIGger:{A|B}:LOGIc:CLAss and TRIGger:{A|B}:PULse:CLAss respectively. This command is similar to selecting Event Trigger Setup from the Trig menu and then selecting the desired Trigger Type.

Some trigger types are not available on some instruments.

Group Trigger

Syntax

```
TRIGger:{A|B}:TYPe {EDGE|LOGIc|PULse|VIDeo|
I2C|CAN|SPI|COMMunication|SERIAL|RS232}}
```

NOTE. *Arguments* `EDGE`, `LOGIc`, `PULse`, `VIDeo`, `I2C`, `CAN`, `SPI`, `COMMunication`, `SERIAL`, *or* `RS232` *are supported by A trigger events.*

NOTE. *Arguments* `EDGE`, `LOGIc`, *or* `PULse` *are supported by B trigger events.*

```
TRIGger:{A|B}:TYPe?
```

Related Commands TRIGger:{A|B}:EDGE?, TRIGger:{A|B}:LOGIc:CLAss, TRIGger:{A|B}: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.

`VIDeo` specifies that the trigger occurs when a video signal is found. Requires an instrument with video hardware.

I2C specifies that a trigger occurs when an Inter-IC Control signal is found.

CAN specifies that a trigger occurs when a Controller Area Network frame signal is found.

SPI specifies that a trigger occurs when a Serial Peripheral Interface signal is found.

COMMunication (Option MTM) specifies that a trigger occurs when a communications signal is found. Supports AMI, HDB3, B3ZS, B6ZS, B8ZS, CMI, MLT3, Manchester, and NRZ encoded communications signals. COMMunication is available only if Option MTM is installed.

SERIAL specifies that a trigger occurs when NRZ-encoded data is found, providing a 32-bit serial word. This argument is available with instruments with Option PTM.

RS232 takes a signal on a data source and allows you to trigger on data within the RS232 bitstream. The data is only one byte wide.

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|B}:UPPerthreshold:CH<x>

This command sets or queries the CH<x> upper trigger level for TRIGger:LVLSrcpreference SRCDependent. The CH<x> range is 1 to 4.

Group

Trigger

Syntax

```
TRIGger:{A|B}:UPPerthreshold:CH<x> {ECL|TTL|<NR3>}
TRIGger:{A|B}:UPPerthreshold:CH<x>?
```

Related Commands

TRIGger:LVLSrcpreference, TRIGger:{A|B}:LOWerthreshold:CH<x>

Arguments

ECL specifies the ECL high level.

TTL specifies the TTL high level.

<NR3> specifies the trigger level in user units (usually volts).

Examples

TRIGGER:A:UPPERTHRESHOLD:CH1 TTL This command sets the A trigger level for Channel 1 to TTL high level.

TRIGGER:A:UPPERTHRESHOLD:CH2? might return :TRIGGER:A:UPPERTHRESHOLD:CH2 1.3000E+00, indicating that the A trigger level for Channel 2 is set to 1.3 V.

TRIGger:A

This command 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 is equivalent to pushing the LEVEL knob on the front panel.

Group Trigger

Syntax

```
TRIGger:A SETLevel
TRIGger:A?
```

Related Commands TRIGger:{A|B}:EDGE?, TRIGger:{A|B}:LOGIc?, TRIGger:{A|B}:PULse?

Arguments SETLevel sets the trigger level to 50% of the range of the minimum and maximum values of the trigger input signal.

Examples TRIGGER:A SETLEVEL sets the trigger level to 50% of the range of the minimum and maximum values of the trigger input signal.

TRIGGER:A? might return the following trigger parameters:

```
:TRIGGER:A:MODE AUTO;TYPE EDGE;LEVEL 0.0000;HOLDOFF:BY
DEFAULT;TIME 1.5000E-6;:TRIGGER:A:EDGE:SOURCE
CH1;COUPLING DC;SLOPE RISE;:TRIGGER:A:LOGIC:CLASS
PATTERN;FUNCTION AND;THRESHOLD:CH1 1.2000;CH2
1.2000;CH3 1.2000;CH4 1.2000;:TRIGGER:A:LOGIC:INPUT:CH1
HIGH;CH2 X;CH3 X;:TRIGGER:A:LOGIC:PATTERN:INPUT:CH4
X;:TRIGGER:A:LOGIC:PATTERN:WHEN
TRUE;WHEN:LESSLIMIT 5.0000E-9;MORELIMIT
5.0000E-9;:TRIGGER:A:LOGIC:SETHOLD:CLOCK:EDGE RISE;THRESHOLD
1.2000;SOURCE CH2;:TRIGGER:A:LOGIC:SETHOLD:DATA:THRESHOLD
1.2000;SOURCE CH1;:TRIGGER:A:LOGIC:SETHOLD:HOLDTIME
2.0000E-9;SETTIME 3.0000E-9;:TRIGGER:A:LOGIC:STATE:INPUT:CH4
RISE;:TRIGGER:A:LOGIC:STATE:WHEN
TRUE;:TRIGGER:A:PULSE:CLASS GLITCH;SOURCE
CH1;GLITCH:WIDTH 2.0000E-9;TRIGIF ACCEPT;POLARITY
POSITIVE;:TRIGGER:A:PULSE:WINDOW:TYPE INSIDE;WHEN
OCCURS;WIDTH 2.0000E-9;LOGIC:INPUT:CH1 HIGH;CH2 HIGH;CH3
```

```
HIGH;CH4 HIGH;:TRIGGER:A:PULSE:WINDOW:LOGIC:THRESHOLD:CH1
1.2000;CH2 1.2000;CH3 1.2000;CH4
1.2000;:TRIGGER:A:PULSE:WINDOW:THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:RUNT:LOGIC:THRESHOLD:CH1
1.2000;CH2 1.2000;CH3 1.2000;CH4
1.2000;:TRIGGER:A:PULSE:RUNT:POLARITY
POSITIVE;THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:RUNT:WHEN OCCURS;WIDTH
2.0000E-9;:TRIGGER:A:PULSE:TRANSITION:DELTATIME
2.0000E-9;POLARITY POSITIVE;THRESHOLD:HIGH 1.2000;LOW
800.0000E-3;:TRIGGER:A:PULSE:TRANSITION:WHEN
FASTERTHAN;:TRIGGER:A:PULSE:WIDTH:LOWLIMIT
2.0000E-9;HIGHLIMIT 2.0000E-9;WHEN WITHIN;POLARITY
POSITIVE;:TRIGGER:A:PULSE:TIMEOUT:POLARITY
STAYSHIGH;TIME 2.0000E-9;:TRIGGER:A:VIDEO:CUSTOM:FORMAT
INTERLACED;SCAN RATE1;:TRIGGER:A:VIDEO:FIELD
ALLFIELDS;HOLDOFF:FIELD 0.0000;:TRIGGER:A:VIDEO:LINE
1;POLARITY NORMAL;SCAN RATE1;SOURCE CH1;STANDARD
NTSC;:TRIGGER:A:COMMUNICATION:STANDARD CUSTOM;CODE
NRZ;BITRATE 1544000;AMI:PULSEFORM PLUSONE;THRESHOLD:HIGH
0.0000;LOW 0.0000;:TRIGGER:A:COMMUNICATION:CLOCK:POLARITY
RISE;:TRIGGER:A:COMMUNICATION:CMI:PULSEFORM
PLUSONE;:TRIGGER:A:COMMUNICATION:SOURCE CH1;SOURCE:TYPE DATA.
```

TRIGger:A:BUS:CAN:CONDition

This command sets or queries the CAN bus trigger condition.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:CAN:CONDition
{ERRor|DATA|IDANDDATA|EOF|IDentifier|ACKMISS|SOF|FRAMEtype}
TRIGger:A:BUS:CAN:CONDition?
```

Arguments Arguments are the available trigger conditions.

Examples `TRIGger:A:BUS:CAN:CONDition EOF` sets the trigger condition to end of frame.

TRIGger:A:BUS:CAN:CONDition? might return :TRIGGER:A:BUS:CAN:CONDITION SOF, indicating that the trigger condition is set to a start of frame.

TRIGger:A:BUS:CAN:DATa:DIRection

This command sets or queries the CAN bus trigger data direction.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:CAN:DATa:DIRection {DONTCare|READ|WRITE}
TRIGger:A:BUS:CAN:DATa:DIRection?
```

Arguments

DONTCare specifies the direction is a don't care.

READ specifies the read direction.

WRITE specifies the write direction.

Examples

TRIGger:A:BUS:CAN:DATa:DIRection READ sets the data direction to read.

TRIGger:A:BUS:CAN:DATa:DIRection? might return :TRIGGER:A:BUS:CAN:DATA:DIRECTION DONTCARE indicating the data direction is don't care.

TRIGger:A:BUS:CAN:DATa:FORMat

This command sets or queries the CAN bus trigger data format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:CAN:DATa:FORMat {BINary|HEX}
TRIGger:A:BUS:CAN:DATa:FORMat?
```

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples `TRIGger:A:BUS:CAN:DATa:FORMat BINARY` sets the data format to binary.

`TRIGger:A:BUS:CAN:DATa:FORMat?` might return `:TRIGGER:A:BUS:CAN:DATA:FORMAT BINARY` indicating the data format is set to binary.

TRIGger:A:BUS:CAN:DATa:QUALifier

This command sets or queries CAN bus trigger data qualifier.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN:DATa:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
TRIGger:A:BUS:CAN:DATa:QUALifier?
```

Arguments Arguments are the available data qualifiers.

Examples `TRIGger:A:BUS:CAN:DATa:QUALifier UNEQ` sets the data qualifier to unequal.

`TRIGger:A:BUS:CAN:DATa:QUALifier?` might return `:TRIGGER:A:BUS:CAN:DATA:QUALIFIER EQUAL` indicating that the data qualifier is equal.

TRIGger:A:BUS:CAN:DATa:SIZe

This command sets or queries the CAN bus trigger data size (the number of contiguous data bytes to trigger on).

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN:DATa:SIZe <nr3>
TRIGger:A:BUS:CAN:DATa:SIZe?
```

Arguments `<nr3>` specifies the data size.

Examples `TRIGger:A:BUS:CAN:DATa:SIZe 2` sets the data size to 2 bytes.

`TRIGger:A:BUS:CAN:DATa:SIZe?` might return `:TRIGGER:A:BUS:CAN:DATA:SIZE 1` indicating that the data size is 1 byte.

TRIGger:A:BUS:CAN:DATa:VALue

This command sets or queries CAN bus trigger data value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN:DATa:VALue <string>
TRIGger:A:BUS:CAN:DATa:VALue?
```

Arguments `<string>` specifies the data value.

Examples `TRIGger:A:BUS:CAN:DATa:VALue 1010` sets the data value to XXXX1010.

`TRIGger:A:BUS:CAN:DATa:VALue?` might return `:TRIGGER:A:BUS:CAN:DATA:VALUE "XXXX1010"`, indicating the data value is "XXXX1010".

TRIGger:A:BUS:CAN:FRAMEtype

This command sets or queries CAN bus trigger frame type.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN:FRAMEtype {DATa|ERRor|OVERLoad|REMote}
TRIGger:A:BUS:CAN:FRAMEtype?
```

Arguments Arguments are the available frame types.

Examples `TRIGger:A:BUS:CAN:FRAMEtype ERROR` sets the frame type to error.

`TRIGger:A:BUS:CAN:FRAMEtype?` might return `:TRIGGER:A:BUS:CAN:FRAMETYPE DATA`, indicating the frame type is data.

TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:DIRection

This command sets or queries the CAN bus trigger identifier direction.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:DIRection
{DONTCare|READ|WRITE}
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:DIRection?
```

Arguments Arguments are the available identifier directions.

Examples `TRIGger:A:BUS:CAN:IDentifier:DIRection READ` sets the identifier direction to read.

`TRIGger:A:BUS:CAN:IDentifier:DIRection?` might return `:TRIGGER:A:BUS:CAN:IDENTIFIER:DIRECTION DONTCARE`, indicating the identifier direction is don't care.

TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:FORMat

This command sets or queries the CAN bus identifier format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:FORMat {BINary|HEX}
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:FORMat?
```

Arguments `BINary` specifies the identifier format as binary.

`HEX` specifies the identifier format as hexadecimal.

Examples `TRIGger:A:BUS:CAN:IDentifier:FORMat HEX` sets the identifier format to hexadecimal.

`TRIGger:A:BUS:CAN:IDentifier:FORMat?` might return `:TRIGGER:A:BUS:CAN:IDENTIFIER:FORMAT BINARY`, indicating the identifier format is binary.

TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:MODe

This command sets or queries CAN bus trigger identifier mode.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:MODe
{EXTENDed|STandard}
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:MODe?
```

Arguments `EXTENDed` specifies the extended identifier mode.

`STandard` specifies the standard identifier mode.

Examples `TRIGger:A:BUS:CAN:IDentifier:MODe EXTEND` sets the identifier mode to extended.

`TRIGger:A:BUS:CAN:IDentifier:MODe?` might return `:TRIGGER:A:BUS:CAN:IDENTIFIER:MODE STANDARD`, indicating the identifier mode is set to standard.

TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:VALue

This command sets or queries the CAN bus trigger identifier value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:VALue <string>
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:VALue?
```

Arguments <string> specifies the identifier value.

Examples `TRIGger:A:BUS:CAN:IDentifier:VALue 01` sets the identifier value to XXXXXXXXX01,

`TRIGger:A:BUS:CAN:IDentifier:VALue?` might return `:TRIGGER:A:BUS:CAN:IDENTIFIER:VALUE "XXXXXXXXX01"`, indicating the identifier value is set to 01.

TRIGger:A:BUS:DATa:FORMat

This command sets or queries the trigger data format for the bus.

Group Trigger

Syntax
```
TRIGger:A:BUS:DATa:FORMat {BINary|HEXadecimal|SYMBolic}
TRIGger:A:BUS:DATa:FORMat?
```

Arguments `BINary` specifies the trigger data format to Binary.

`HEXadecimal` specifies the trigger data format to Hexadecimal.

`SYMBolic` specifies the trigger data format as Symbolic.

Examples `TRIGGER:A:BUS:DATA:FORMAT HEX` sets the trigger data format for the bus to HEXadecimal.

`TRIGGER:A:BUS:DATA:FORMAT?` might return `:TRIGGER:A:BUS:DATA:FORMAT BIN`, indicating that the trigger data format for the bus is set to BINary.

TRIGger:A:BUS:DATa:VALue

This command sets or queries the trigger data value depending on the format selected for the bus. The data value varies depending on the bus type and the number of input signals.

Group Trigger

Syntax
```
TRIGger:A:BUS:DATa:VALue <string>
TRIGger:A:BUS:DATa:VALue?
```

Arguments <string> is the data value in binary or hexadecimal format. The valid characters are 0-9, A-F, and X for addresses in hexadecimal format and 0, 1, and X otherwise.

Examples TRIGGER:A:BUS:DATA:VALUE "656565XX65656AF37565" sets the trigger data value for the bus to 656565XX65656AF37565 in hexadecimal format.

TRIGGER:A:BUS:DATA:VALUE? might return :TRIGGER:A:BUS:DATA:VALUE "0100X0XX101", indicating that the trigger data value for the value is set to 0100X0XX101 in binary format.

TRIGger:A:BUS:ETHERnet:CONDition

This command specifies a field or condition within an Ethernet frame to trigger on. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:CONDition
{SFD|MACADDRess|MACLENgth|IPHeader|TCPHeader|DATa|EOP
|FCSError|QTAG|TCPData}
```

```
TRIGger:A:BUS:ETHERnet:CONDition?
```

Related Commands Most of the other TRIGger:A:BUS:B<x>:ETHERnet commands are impacted by the setting of this command.

TRIGger:A:BUS:ETHERnet:QUALifier

Arguments SFD – Start of frame delimiter.

MACADDRess – MAC addresses field.

MACLENgth – MAC length/type field.

IPHeader – IP header field. .

TCPHeader – TCP header field.

DATa – TCP/IPv4 or MAC protocol client data field.

EOP – End of Packet field.

IDLe – Idle field.

FCSError – Frame Check Sequence Error (CRC) field.

QTAG – IEEE 802.1Q (VLAN) control information field.

Examples

TRIGGER:A:BUS:ETHERNET:CONDITION MACADDRESS specifies MACADDRess as the field within an Ethernet frame to trigger on.

TRIGGER:A:BUS:ETHERNET:CONDITION? might return DATA, indicating that DATA is the currently specified field within an Ethernet frame to trigger on.

TRIGger:A:BUS:ETHERnet:DATa:FORMat

This command and query sets the format of the data to either binary or hex.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:DATa:FORMat {BINARY | HEX}
TRIGger:A:BUS:ETHERnet:DATa:FORMat?
```

Arguments

BINARY sets the format to binary.

HEX sets the format to hexadecimal.

Returns

TRIGger:A:BUS:ETHERnet:DATa:FORMat? might return :TRIGGER:A:BUS:ETHERNET:DATA:FORMAT BINARY indicating the Ethernet data format is set to binary.

TRIGger:A:BUS:ETHERnet:DATa:OFFSet

This command specifies the data offset value, in bytes, to use when triggering on the Ethernet data field. The trigger condition needs to be set to DATa. (MSO/DPO5000/B series only.)

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:DATa:OFFSet <NR1>
TRIGger:A:BUS:ETHERnet:DATa:OFFSet?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:DATa:SIZe

TRIGger:A:BUS:ETHERnet:DATa:VALue

Arguments <NR1> is an integer whose minimum and default values are -1 (don't care) and maximum is 1,499.

Examples TRIGGER:A:BUS:ETHERNET:DATA:OFFSET 36 sets the data offset to 36 bytes.

TRIGGER:A:BUS:ETHERNET:DATA:OFFSET? might return -1, indicating that the data offset value is the default value, -1, meaning “don’t care”.

TRIGger:A:BUS:ETHERnet:DATa:SIZe

This command specifies the number of contiguous TCP/IPv4/MAC client bytes to use when triggering on the Ethernet data field. The default is all X’s (don’t care). The trigger condition needs to be set to DATa. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:DATa:SIZe <NR1>
TRIGger:A:BUS:ETHERnet:DATa:SIZe?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:QUALifier

Arguments The minimum and default values are 1, and the maximum value is 16.

Examples TRIGGER:A:BUS:ETHERNET:DATA:SIZE 4 specifies 4 as the number of contiguous TCP/IPv4/MAC client data bytes to use when triggering on the Ethernet data field.

TRIGGER:A:BUS:ETHERNET:DATA:SIZE? might return 6, indicating that 6 is the currently specified number of contiguous TCP/IPv4/MAC client data bytes to use when triggering on the Ethernet data field.

TRIGger:A:BUS:ETHERnet:DATa:VALue

This command specifies the binary value to use when triggering on the Ethernet binary data field. The default is all X's (don't care). The trigger condition needs to be set to `DATa`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:DATa:VALue <QString>
TRIGger:A:BUS:ETHERnet:DATa:VALue?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:QUALifier

TRIGger:A:BUS:ETHERnet:DATa:OFFSet

TRIGger:A:BUS:ETHERnet:DATa:SIZe

Arguments QString is a quoted string where the allowable characters are 0, 1, and X. The allowable number of characters depends on the setting for size (using TRIGger:A:BUS:ETHERnet:DATa:SIZe). The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `TRIGGER:A:BUS:ETHERNET:DATA:VALUE "00001000"` specifies 00001000 as the value to use when triggering on the Ethernet binary data field, assuming that `TRIGGER:A:BUS:ETHERNET:DATA:SIZe` is set to 1 byte.

`TRIGGER:A:BUS:ETHERNET:DATA:VALUE?` might return `"00001000"`, indicating 0000100 is the currently specified value used when triggering on the Ethernet binary data field.

TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet IPv4 header address destination field. The trigger condition needs to be set to `IPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax
```
TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat
{BINary|HEX|DECImal}
TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

`DECImal` specifies the data format as decimal.

Examples `TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:FORMAT HEX` specifies `HEX` as the data format to use when triggering on the Ethernet IPv4 header address destination field.

`TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:FORMAT?` might return `BINARY`, indicating binary is the currently specified data format used when triggering on the Ethernet IPv4 header address destination field.

TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue

This command specifies the 32-bit IPv4 value to use when triggering on the Ethernet IPv4 header address destination field. The default is all X's (don't care). The trigger condition needs to be set to `IPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax
```
TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue
<QString>
TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat

Arguments QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:VALUE "11000000101010000000000000000001"` specifies 192.168.0.1 as the value to use when triggering on the Ethernet IPv4 header address destination field.

`TRIGGER:A:BUS:ETHERNET:IPHEADER:DESTINATIONADDR:VALUE?` might return `"11000000101010000000000000000001"`, indicating that 192.168.0.1 is the currently specified value used when triggering on the Ethernet IPv4 header address destination field.

TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat

This command specifies the data format (binary or hex) of the bit pattern to use when triggering on the Ethernet IPv4 header protocol field. The trigger condition needs to be set to `MACADDRess`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax
```
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat {BINary|HEX}
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples `TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:FORMAT HEX` specifies `HEX` as the data format to use when triggering on the Ethernet IPv4 header protocol field.

`TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:FORMAT?` might return `BINARY`, indicating that binary is the currently specified data format used when triggering on the Ethernet IPv4 header protocol field.

TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue

This command specifies the 8-bit binary value to use when triggering on the Ethernet IPv4 header protocol field. The default is all X's (don't care). The trigger condition needs to be set to `IPHeader`. (MSO/DPO5000/B series only.)

NOTE. *Commonly used protocol values are 1 (ICMP), 2 (IGMP), 6 (TCP) and 17 (UDP).*

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue <QString>
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat

Arguments QString is a quoted string of up to 8 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:VALUE "01010010"` specifies 01010010 as the value to use when triggering on the Ethernet IPv4 header protocol field.

`TRIGGER:A:BUS:ETHERNET:IPHEADER:PROTOCOL:VALUE?` might return "XXXXXXXX", indicating that the trigger value has been set to "don't care".

TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat

This command specifies the data format (binary or hex) of the bit pattern to use when triggering on the Ethernet IPv4 header address source field. The trigger condition needs to be set to `IPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat
{BINary|HEX|DECImal}
TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue

Arguments

`BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

`DECImal` specifies the data format as decimal.

Examples

`TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:FORMAT HEX` specifies to use the `HEX` data format when triggering on the Ethernet IPv4 header address source field.

`TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:FORMAT?` might return `BINARY`, indicating that the data format is set to binary.

TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue

This command specifies the 32-bit value to use when triggering on the Ethernet IPv4 header address source field. The default is all X's (don't care). The trigger condition needs to be set to `IPHeader`.

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue <QString>
TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat

Arguments

QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

`TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:VALUE "11000000101010000000000000000001"` specifies to use the value of 192.168.0.1 when triggering on the Ethernet IPv4 header address source field.

`TRIGGER:A:BUS:ETHERNET:IPHEADER:SOURCEADDR:VALUE?` might return `"11000000101010000000000000000001"`, indicating that the trigger value has been set to 192.168.0.1.

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat

This command specifies the data format (binary or hex) of the bit pattern to use when triggering on the Ethernet MAC address destination field. The trigger condition needs to be set to `MACADDRess`. (MSO/DPO5000/B series only.)

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat
{BINary|HEX}
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue

Arguments

`BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples

`TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:FORMAT HEX` specifies to use the `HEX` data format when triggering on the Ethernet MAC address destination field.

`TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:FORMAT?` might return `BINARY`, indicating that the data format is set to binary.

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue

This command specifies the 48-bit value to use when triggering on the Ethernet MAC address destination field. The default is all X's (don't care). The trigger condition needs to be set to `MACADDRess`. (MSO/DPO5000/B series only.)

NOTE. *MAC Addresses are 48-bit values such as 08:00:11:1E:C9:AE hex.*

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue
<QString>
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat

Arguments QString is a quoted string of up to 48 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:VALUE "XXXXXXXX00110101111111000000011110101010110010000"` specifies to use the value of XX:35:FC:07:AA:C8 hex when triggering on the Ethernet MAC address destination field.

`TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:DESTINATION:VALUE?` might return `"XXXXXXXX00110101111111000000011110101010110010000"`, indicating a MAC address destination field value of XX:35:FC:07:AA:C8 hex.

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet MAC address source field. The trigger condition needs to be set to `MACADDRess`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat
{BINary|HEX}
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue

Arguments

`BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples

`TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:FORMAT HEX` specifies to use the `HEX` data format when triggering on the Ethernet MAC address source field.

`TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:FORMAT?` might return `BINARY`, indicating that the data format is set to binary.

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue

This command specifies the 48-bit value to use when triggering on the Ethernet MAC address source field. The default is all X's (don't care). The trigger condition needs to be set to `MACADDRess`. (MSO/DPO5000/B series only.)

NOTE. *MAC Addresses are 48-bit values such as 08:00:11:1E:C9:AE hex.*

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue <QString>
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat

Arguments

QString is a quoted string of up to 48 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

`TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:VALUE "XXXXXXXX001101011111110000000111101010101100100 0"` specifies to use the value of XX:35:FC:07:AA:C8 hex when triggering on the Ethernet MAC address source field.

TRIGGER:A:BUS:ETHERNET:MAC:ADDRESS:SOURCE:VALUE? might return "XXXXXXXX0011010111111100000001111010101011001000", indicating a MAC address source field value of XX:35:FC:07:AA:C8 hex.

TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet MAC length or type. The trigger condition needs to be set to MACADDRess. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat
{BINary|HEX}
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples TRIGGER:A:BUS:ETHERNET:MAC:LENGTH:FORMAT HEX specifies to use the HEX data format when triggering on the Ethernet MAC length.

TRIGGER:A:BUS:ETHERNET:MAC:LENGTH:FORMAT? might return BINARY, indicating that the data format to use when triggering on the Ethernet MAC length is set to binary.

TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue

This command specifies the 16-bit value to use when triggering on the Ethernet MAC length/type. The default is all X's (don't care). The trigger condition needs to be set to MACADDRess. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax
```
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue <QString>
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat

Arguments QString is a quoted string of up to 16 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `TRIGGER:A:BUS:ETHERNET:MAC:TYPE:VALUE "XXXXXXXX00001000"` specifies to use the hexadecimal value XX08 when triggering on the Ethernet MAC type.

`TRIGGER:A:BUS:ETHERNET:MAC:TYPE:VALUE?` might return "XXXXXXXX00001000", indicating an Ethernet MAC type value of XX08 hex.

TRIGger:A:BUS:ETHERnet:QTAG:FORMat

This command specifies the data format (binary or hex) to use when triggering on the Q-Tag field. The trigger condition needs to be set to `QTAG`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax
```
TRIGger:A:BUS:ETHERnet:QTAG:FORMat {BINary|HEX}
TRIGger:A:BUS:ETHERnet:QTAG:FORMat?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:QTAG:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples

`TRIGGER:A:BUS:ETHERNET:QTAG:FORMAT HEX` specifies to use the `HEX` data format when triggering on the Ethernet Q-Tag field.

`TRIGGER:A:BUS:ETHERNET:QTAG:FORMAT?` might return `BINARY`, indicating that the data format to use for triggering on the Ethernet Q-Tag field has been set to binary.

TRIGger:A:BUS:ETHERnet:QTAG:VALue

This command specifies the 32-bit value to use when triggering on the Ethernet Q-Tag field. The default is all X's (don't care). The trigger condition needs to be set to `QTAG`. (MSO/DPO5000/B series only.)

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:QTAG:VALue <QString>
TRIGger:A:BUS:ETHERnet:QTAG:VALue?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:QTAG:FORMat

Arguments

QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

`TRIGGER:A:BUS:ETHERNET:QTAG:VALUE` "XXXXXXXXXXXXXXXXXXXX010010001010" specifies to use the value of hexadecimal XXXXX48A when triggering on the Ethernet Q-Tag field.

`TRIGGER:A:BUS:ETHERNET:QTAG:VALUE?` might return "XXXXXXXXXXXXXXXXXXXX010010001010", indicating that hexadecimal XXXXX48A has been set as the Ethernet Q-Tag field trigger value.

TRIGger:A:BUS:ETHERnet:QUALifier

This command specifies the qualifier to be used with the Ethernet trigger condition is set to `MACLENgth` or `DATa`. Normally, the Ethernet qualifier is set to "equal to". (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax
```
TRIGger:A:BUS:ETHERnet:QUALifier
{LESSthan|MOREthan|EQual|UNEQual|LESSEQual|MOREEQual}

TRIGger:A:BUS:ETHERnet:QUALifier?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

Arguments

`LESSthan` sets the qualifier to less than.

`MOREthan` sets the qualifier to greater than.

`EQual` sets the qualifier to equal.

`UNEQual` sets the qualifier to not equal.

`LESSEQual` sets the qualifier to less than or equal.

`MOREEQual` sets the qualifier to greater than or equal.

Examples `TRIGGER:A:BUS:ETHERNET:QUALIFIER LESSTHAN` sets the qualifier to “less than”.

TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header acknowledgement number. The trigger condition needs to be set to `TCPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax
```
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat
{BINary|HEX|DECImal}
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

DECImal specifies the data format as decimal.

Examples

TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:FORMAT HEX specifies to use the HEX data format when triggering on the Ethernet TCP header acknowledgement number.

TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:FORMAT? might return BINARY, indicating that the data format to use when triggering on the Ethernet TCP header acknowledgement number has been set to binary.

TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue

This command specifies the 32-bit value to use when triggering on the Ethernet TCP header acknowledgement number. The default is all X's (don't care). The trigger condition needs to be set to TCPHeader. (MSO/DPO5000/B series only.)

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue <QString>
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat

Arguments

QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:VALUE "XXXXXXXXXXXXXXXXXXXX00001000" specifies hexadecimal XXXXXX08 as the value to use when triggering on the Ethernet TCP header acknowledgement number.

TRIGGER:A:BUS:ETHERNET:TCPHEADER:ACKNUM:VALUE? might return "XXXXXXXXXXXXXXXXXXXX00001000", indicating that hexadecimal XXXXXX08 has been specified as the value to use when triggering on the Ethernet TCP header acknowledgement number.

TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header destination port number. The trigger condition needs to be set to `TCPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat
{BINary|HEX|DECImal}
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

`DECImal` specifies the data format as decimal.

Examples `TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:FORMAT HEX` specifies to use the `HEX` data format when triggering on the Ethernet TCP header destination port number.

`TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:FORMAT?` might return `BINARY`, indicating that the data format to use when triggering on the Ethernet TCP header destination port number has been set to binary.

TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue

This command specifies the 16-bit value to use when triggering on the Ethernet TCP header destination port number. The default is all X's (don't care). The trigger condition needs to be set to `TCPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue
<QString>
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat

Arguments

QString is a quoted string of up to 16 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

`TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:VALUE "XXXXXXXX00100010"` specifies to use the value of hexadecimal XX22 when triggering on the Ethernet TCP header destination port number.

`TRIGGER:A:BUS:ETHERNET:TCPHEADER:DESTINATIONPORT:VALUE?` might return "XXXXXXXX00100010", indicating that hexadecimal XX22 has been set as the value to use when triggering on the Ethernet TCP header destination port number.

TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header sequence number. The trigger condition needs to be set to `TCPHeader`. (MSO/DPO5000/B series only.)

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat
{BINary|HEX|DECImal}
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue

Arguments

`BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

DECImal specifies the data format as decimal.

Examples

TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:FORMAT HEX specifies to use the HEX data format when triggering on the Ethernet TCP header sequence number.

TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:FORMAT? might return BINARY, indicating that the binary data format has been specified to use when triggering on the Ethernet TCP header sequence number.

TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue

This command specifies the 32-bit value to use when triggering on the Ethernet TCP header sequence number. The default is all X's (don't care). The trigger condition needs to be set to TCPHeader. (MSO/DPO5000/B series only.)

Conditions

Requires the SR-ENET Triggering and Analysis application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue <QString>
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat

Arguments

QString is a quoted string of up to 32 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:VALUE "XXXXXXXXXXXXXXXXXXXX000100010001" specifies to use the value of hexadecimal XXXXX111 when triggering on the Ethernet TCP header sequence number.

TRIGGER:A:BUS:ETHERNET:TCPHEADER:SEQNUM:VALUE? might return "XXXXXXXXXXXXXXXXXXXX000100010001", indicating that hexadecimal XXXXX111 has been specified as the value to use when triggering on the Ethernet TCP header sequence number.

TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the Ethernet TCP header source port number. The trigger condition needs to be set to `TCPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat
{BINary|HEX|DECImal}
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat?
```

Related Commands TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

`DECImal` specifies the data format as decimal.

Examples `TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:FORMAT HEX` specifies to use the `HEX` data format when triggering on the Ethernet TCP header source port number.

`TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:FORMAT?` might return `BINARY`, indicating that binary is the currently specified data format for triggering on the Ethernet TCP header source port number.

TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue

This command specifies the 16-bit value to use when triggering on the Ethernet TCP header source port number. The default is all X's (don't care). The trigger condition needs to be set to `TCPHeader`. (MSO/DPO5000/B series only.)

Conditions Requires the SR-ENET Triggering and Analysis application.

Group Trigger

Syntax

```
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue <QString>
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue?
```

Related Commands

TRIGger:A:BUS:ETHERnet:CONDition

TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat

Arguments

QString is a quoted string of up to 16 characters where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples

`TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:VALUE "XXXX000010100110"` specifies to use the value of hexadecimal X0A6 when triggering on the Ethernet TCP header source port number.

`TRIGGER:A:BUS:ETHERNET:TCPHEADER:SOURCEPORT:VALUE?` might return "XXXXX01001010110", indicating that hexadecimal X0A6 has been specified as the value to use when triggering on the Ethernet TCP header source port number.

TRIGger:A:BUS:FLEXRAY:CONDition

This command sets or queries the FLEXRAY bus trigger condition.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:CONDition
{SOF|FRAMEtype|IDentifier|CYCLEcount|HEADer|DATA|IDANDDATA|EOF|ERRor
TRIGger:A:BUS:FLEXRAY:CONDition?
```

Arguments

Arguments are the available trigger conditions.

Examples

`TRIGger:A:BUS:FLEXRAY:CONDition EOF` sets the trigger condition to end of frame.

`TRIGger:A:BUS:FLEXRAY:CONDition?` might return `:TRIGGER:A:BUS:FLEXRAY:CONDITION SOF`, indicating the trigger condition is start of frame.

TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat

This command sets or queries FLEXRAY bus trigger cycle count format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat {BINary|HEX}
TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat?
```

Arguments `BINary` specifies the cycle count format as binary.

`HEX` specifies the cycle count format as hexadecimal.

Examples `TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat HEX` sets the cycle count format to hexadecimal.

`TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat?` might return `:TRIGGER:A:BUS:FLEXRAY:CYCLECOUNT:FORMAT BINARY`, indicating the cycle count format is binary.

TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier

This command sets or queries FLEXRAY bus trigger cycle count qualifier.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier?
```

Arguments Argument are the available cycle count qualifiers.

Examples `TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier LESSTHAN` sets the cycle count qualifier to less than.

TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier? might return :TRIGGER:A:BUS:FLEXRAY:CYCLECOUNT:QUALIFIER EQUAL, indicating the cycle count qualifier is equal.

TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue

This command sets or queries the FLEXRAY bus trigger cycle count value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue <STRING>
TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue?
```

Arguments <STRING> specifies the cycle count value.

Examples TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue "1010" sets the cycle count value to XX1010.

TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue? might return :TRIGGER:A:BUS:FLEXRAY:CYCLECOUNT:VALUE "XXXXXX", indicating the cycle count value has not been set.

TRIGger:A:BUS:FLEXRAY:DATa:FORMat

This command sets or queries the FLEXRAY bus trigger data format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:DATa:FORMat {BINary|HEX}
TRIGger:A:BUS:FLEXRAY:DATa:FORMat?
```

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples `TRIGger:A:BUS:FLEXRAY:DATa:FORMat HEX` sets the data format to hexadecimal.

`TRIGger:A:BUS:FLEXRAY:DATa:FORMat?` might return `:TRIGGER:A:BUS:FLEXRAY:DATA:FORMAT BINARY`, indicating the data format is binary.

TRIGger:A:BUS:FLEXRAY:DATa:OFFSet

This command sets or queries the FLEXRAY bus trigger data offset.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:DATa:OFFSet <NR3>
TRIGger:A:BUS:FLEXRAY:DATa:OFFSet?
```

Arguments `<NR3>` specifies the data offset in bytes.

Examples `TRIGger:A:BUS:FLEXRAY:DATa:OFFSet 1` sets the data offset to 1 byte.

`TRIGger:A:BUS:FLEXRAY:DATa:OFFSet?` might return `:TRIGGER:A:BUS:FLEXRAY:DATA:OFFSET -1`, indicating the data offset is don't care.

TRIGger:A:BUS:FLEXRAY:DATa:QUALifier

This command sets or queries the FLEXRAY bus trigger data qualifier.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:DATa:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
TRIGger:A:BUS:FLEXRAY:DATa:QUALifier?
```

Arguments Arguments are the available data qualifiers.

Examples `TRIGger:A:BUS:FLEXRAY:DATa:QUALifier LESSTHAN` sets the data qualifier to less than.

`TRIGger:A:BUS:FLEXRAY:DATa:QUALifier?` might return `:TRIGGER:A:BUS:FLEXRAY:DATA:QUALIFIER EQUAL`, indicating the data qualifier is equal.

TRIGger:A:BUS:FLEXRAY:DATa:SIZe

This command sets or queries the FLEXRAY bus trigger data size in bytes.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:DATa:SIZe <NR3>
TRIGger:A:BUS:FLEXRAY:DATa:SIZe?
```

Arguments `<NR3>` specifies the data size in bytes. A setting of X/Don't care is accomplished by setting the size to –1.

Examples `TRIGger:A:BUS:FLEXRAY:DATa:SIZe 2` sets the data size to 2 bytes.

`TRIGger:A:BUS:FLEXRAY:DATa:SIZe?` might return `:TRIGGER:A:BUS:FLEXRAY:DATA:SIZE 1`, indicating the data size is 1 byte.

TRIGger:A:BUS:FLEXRAY:DATa:VALue

This command sets or queries the FLEXRAY bus trigger data value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:DATa:VALue <STRING>
TRIGger:A:BUS:FLEXRAY:DATa:VALue?
```

Arguments `<STRING>` specifies the data value.

Examples

`TRIGger:A:BUS:FLEXRAY:DATa:VALue “1010”` sets the data value to XXXX1010.

`TRIGger:A:BUS:FLEXRAY:DATa:VALue?` might return `:TRIGGER:A:BUS:FLEXRAY:DATA:VALUE "XXXXXXXX"`, indicates the data value is don’t care.

TRIGger:A:BUS:FLEXRAY:EOFTYPE

This command sets or queries the FLEXRAY bus trigger end of file type.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:EOFTYPE {ALL|STATIC|DYNAMIC}
TRIGger:A:BUS:FLEXRAY:EOFTYPE?
```

Arguments

`ALL` specifies either end of file type.

`STATIC` specifies the static end of file type.

`DYNAMIC` specifies the static end of file type.

Examples

`TRIGger:A:BUS:FLEXRAY:EOFTYPE STATIC` sets the end of file type to static.

`TRIGger:A:BUS:FLEXRAY:EOFTYPE?` might return `:TRIGGER:A:BUS:FLEXRAY:EOFTYPE ALL`, indicating the end of file type static or dynamic.

TRIGger:A:BUS:FLEXRAY:ERRTYPE

This command sets or queries the FLEXRAY bus trigger error type.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:ERRTYPE
{CRCHEADer|CRCTRAILer|NULLFRSTATIC|NULLFRDYNAMIC
|SYNCFRAME|STARTUPNOSYNC}
TRIGger:A:BUS:FLEXRAY:ERRTYPE?
```

Arguments Arguments are the available error types.

Examples `TRIGger:A:BUS:FLEXRAY:ERRTYPE SYNCFRAME` sets the error type to SYNCFRAME.

`TRIGger:A:BUS:FLEXRAY:ERRTYPE?` might return `:TRIGGER:A:BUS:FLEXRAY:ERRTYPE CRCHEADER`, indicating the error type is set to CRCHEADER.

TRIGger:A:BUS:FLEXRAY:FRAMEType

This command sets or queries the FLEXRAY bus trigger frame type.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:FRAMEType
{NORMal|PAYload|NULL|SYNC|STARTup}
TRIGger:A:BUS:FLEXRAY:FRAMEType?
```

Arguments Argument are the available frame types.

Examples `TRIGger:A:BUS:FLEXRAY:FRAMEType NULL` sets the frame type to NULL.

`TRIGger:A:BUS:FLEXRAY:FRAMEType?` might return `:TRIGGER:A:BUS:FLEXRAY:FRAMETYPE NORMAL`, indicating the frame type is normal.

TRIGger:A:BUS:FLEXRAY:HEADER:CRC

This command sets or queries the FLEXRAY bus trigger header CRC.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:HEADER:CRC <string>
TRIGger:A:BUS:FLEXRAY:HEADER:CRC?
```

Arguments <string> specifies the cyclic redundancy code.

Examples TRIGger:A:BUS:FLEXRAY:HEADER:CRC “1011” sets the CRC to XXXXXXX1011.

TRIGger:A:BUS:FLEXRAY:HEADER:CRC? might return :TRIGGER:A:BUS:FLEXRAY:HEADER:CRC "XXXXXXXXXXX", indicating the CRC is a don’t care.

TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount

This command sets or queries the FLEXRAY bus trigger header cycle count.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount <string>
TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount?
```

Arguments <string> specifies the header cycle count.

Examples TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount “1010” sets the header cycle count to XX1010.

TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount? might return :TRIGGER:A:BUS:FLEXRAY:HEADER:CYCLECOUNT "XXXXXX", indicating the cycle count is not set.

TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID

This command sets or queries the FLEXRAY bus trigger header frame ID.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID <string>
TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID?
```

Arguments <string> specifies the header frame ID.

Examples TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID "1010" sets the header frame ID to XXXXXXX1010.

TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID? might return :TRIGGER:A:BUS:FLEXRAY:HEADER:FRAMEID "XXXXXXXXXXX", indicating the header frame ID is don't care.

TRIGger:A:BUS:FLEXRAY:HEADER:INDBits

This command sets or queries the FLEXRAY bus trigger header Indicator Bits.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:HEADER:INDBits <string>
TRIGger:A:BUS:FLEXRAY:HEADER:INDBits?
```

Arguments <string> specifies the header Indicator Bits.

Examples TRIGger:A:BUS:FLEXRAY:HEADER:INDBits "1010" sets the header Indicator Bits to X1010.

TRIGger:A:BUS:FLEXRAY:HEADER:INDBits? might return :TRIGGER:A:BUS:FLEXRAY:HEADER:INDBITS "XXXXX", indicating the Indicator Bits have not been set.

TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth

This command sets or queries the FLEXRAY bus trigger header Payload Length.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth <string>
TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth?
```

Arguments <string> specifies the header Payload Length.

Examples `TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth “1010”` sets the header Payload Length to XXX1010.

`TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth?` might retrun `:TRIGGER:A:BUS:FLEXRAY:HEADER:PAYLENGTH "XXXXXXX"` indicating the header Payload Length has is a don’t care.

TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat

This command sets or queries the FLEXRAY bus trigger identifier format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat {BINary|HEX}
TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat?
```

Arguments `BINary` specifies the identifier format as binary.

`HEX` specifies the identifier format as hexadecimal.

Examples `TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat HEX` sets the identifier format to hexadecimal.

`TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat?` might return `:TRIGGER:A:BUS:FLEXRAY:IDENTIFIER:FORMAT BINARY`, indicating the identifier format is binary.

TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier

This command sets or queries the FLEXRAY bus trigger identifier qualifier.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier?
```

Arguments Arguments are the available qualifiers.

Examples `TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier LESSTHAN` sets the identifier qualifier to less than.

`TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier?` might return `:TRIGGER:A:BUS:FLEXRAY:IDENTIFIER:QUALIFIER EQUAL`, indicating the identifier qualifier is equal.

TRIGger:A:BUS:FLEXRAY:IDentifier:VALue

This command sets or queries the FLEXRAY bus trigger identifier value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:FLEXRAY:IDentifier:VALue <string>
TRIGger:A:BUS:FLEXRAY:IDentifier:VALue?
```

Arguments `<string>` specifies the identifier value.

Examples `TRIGger:A:BUS:FLEXRAY:IDentifier:VALue “1010”` sets the identifier value to XXXXXXX1010.

`TRIGger:A:BUS:FLEXRAY:IDentifier:VALue?` might return `:TRIGGER:A:BUS:FLEXRAY:IDENTIFIER:VALUE "XXXXXXXXXXX"`, indicating the identifier value is don’t care.

TRIGger:A:BUS:I2C:ADDress:FORMat

NOTE. *Not available on some models.*

This command sets or queries the address format for the I2C bus trigger.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:I2C:ADDress:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:I2C:ADDress:FORMat?
```

Related Commands TRIGger:A:BUS:I2C:ADDress:MODe

Arguments `BINary` specifies the address format for the I2C bus trigger as Binary.

`HEXadecimal` specifies the address format for the I2C bus trigger as Hexadecimal.

Examples `TRIGGER:A:BUS:I2C:ADDRESS:FORMAT BIN` sets the address format for the I2C bus trigger to BINary.

`TRIGGER:A:BUS:I2C:ADDRESS:FORMAT?` might return `:TRIGGER:A:BUS:I2C:ADDRESS:FORMAT HEX`, indicating that the address format for the I2C bus trigger is set to HEXadecimal.

TRIGger:A:BUS:I2C:ADDress:MODe

NOTE. *Not available on some models.*

This command sets or queries the address mode for the I2C bus trigger.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:I2C:ADDress:MODe {ADDR7|ADDR10}
TRIGger:A:BUS:I2C:ADDress:MODe?
```

Arguments `ADDR7` specifies the address mode as ADDR7.

`ADDR10` specifies the address mode as ADDR10.

Examples `TRIGGER:A:BUS:I2C:ADDRESS:MODE ADDR7` sets the address mode for the I2C bus trigger to ADDR7.

`TRIGGER:A:BUS:I2C:ADDRESS:MODE?` might return `:TRIGGER:A:BUS:I2C:ADDRESS:MODE ADDR10`, indicating that the address mode for the I2C bus trigger is set to ADDR10.

TRIGger:A:BUS:I2C:ADDress:TYPe

NOTE. *Not available on some models.*

This command sets or queries the address type for the I2C bus trigger.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:I2C:ADDress:TYPe
{NONe|STARtbyte|HSMODe|GENeralcall|EEPROm|CBUS}
TRIGger:A:BUS:I2C:ADDress:TYPe?
```

Arguments

`NONe` specifies the address type as None.

`STARtbyte` specifies the address type as Start byte.

`HSMODe` specifies the address type as High Speed mode

`GENeralcall` specifies the address type as General Call.

`EEPROm` specifies the address type as EEPROM.

`CBUS` specifies the address type as CBUS.

Examples

`TRIGGER:A:BUS:I2C:ADDRESS TYPE CBUS` sets the address type for the I2C bus trigger to CBUS.

`TRIGGER:A:BUS:I2C:ADDRESS TYPE?` might return `:TRIGGER:A:BUS:I2C:ADDRESS:TYPE GENERALCALL`, indicating that the address type for the bus trigger is set to GENeralcall.

TRIGger:A:BUS:I2C:ADDress:VALue

NOTE. *Not available on some models.*

This command sets or queries the binary address string used for the I2C trigger of the bus.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:I2C:ADDress:VALue <string>
TRIGger:A:BUS:I2C:ADDress:VALue?
```

Arguments <string> specifies the address value. This is either 7 bits or 10 bits depending on the address mode. The valid characters are 0-9, A-F, and X for addresses in hexadecimal format and 0, 1, and X otherwise.

Examples TRIGGER:A:BUS:I2C:ADDRESS:VALUE "011XX11" sets the 7-bit address value to "011XX11".

TRIGGER:A:BUS:I2C:ADDRESS:VALUE? might return :TRIGGER:A:BUS:I2C:ADDRESS:VALUE "A45X1CX", indicating that the 7-bit address value is set to "A45X1CX".

TRIGger:A:BUS:I2C:CONDition

NOTE. *Not available on some models.*

This command sets or queries the trigger condition for the I2C bus.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:I2C:CONDition
{ACKMISS|ADDress|ADDRANDDATA|DATa|REPEATstart|STARt|STOP}
TRIGger:A:BUS:I2C:CONDition?
```

Arguments ACKMISS sets the trigger condition to Missing Acknowledgement.

ADDRess sets the trigger condition to Address.

ADDRANDDATA sets the trigger condition to Address and Data.

DATA sets the trigger condition to data.

REPEATstart sets the trigger condition to Repeat of Start.

STARt sets the trigger condition to Start.

STOP sets the trigger condition to Stop.

Examples TRIGGER:A:BUS:I2C:CONDITION START sets the trigger condition for the I2C bus to start.

TRIGGER:A:BUS:I2C:CONDITION? might return :TRIGGER:A:BUS:I2C:CONDITION STOP, indicating that the trigger condition for the I2C bus is set to stop.

TRIGger:A:BUS:I2C:DATa:DIRection

NOTE. *Not available on some models.*

This command sets or queries the data direction for the I2C bus.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:I2C:DATa:DIRection {DONTCare|READ|WRITE}
TRIGger:A:BUS:I2C:DATa:DIRection?
```

Related Commands TRIGger:A:BUS:I2C:DATa:FORMat

Arguments

`DONTCare` sets the data direction for the I2C bus to Don't Care.

`READ` sets the data direction for the I2C bus to Read.

`WRITE` sets the data direction for the I2C bus to Write.

Examples

`TRIGGER:A:BUS:I2C:DATA:DIRECTION WRITE` sets data direction for the I2C bus to write.

`TRIGGER:A:BUS:I2C:DATA:DIRECTION?` might return `:TRIGGER:A:BUS:I2C:DATA:DIRECTION READ`, indicating that data direction for the I2C bus is set to read.

TRIGger:A:BUS:I2C:DATa:FORMat

NOTE. *Not available on some models.*

This command sets or queries the trigger data format for the I2C bus.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:I2C:DATa:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:I2C:DATa:FORMat?
```

Related Commands TRIGger:A:BUS:I2C:DATa:DIRection, TRIGger:A:BUS:I2C:DATa:SIZe, TRIGger:A:BUS:I2C:DATa:VALue

Arguments

`BINary` specifies the data format for the I2C bus trigger as Binary.

`HEXadecimal` specifies the data format for the I2C bus trigger as Hexadecimal.

Examples

`TRIGGER:A:BUS:I2C:DATA:FORMAT BINARY` sets the data format of the I2C bus trigger to BINary.

`TRIGGER:A:BUS:I2C:DATA:FORMAT?` might return `:TRIGGER:A:BUS:IC2:DATA:FORMAT HEX`, indicating that the data format of the I2C bus trigger is set to HEXadecimal.

TRIGger:A:BUS:I2C:DATa:SIZe

NOTE. *Not available on some models.*

This command sets or queries the length of the data string in bytes to be used for an I2C bus trigger when the trigger condition is Data or Addr + Data.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:I2C:DATa:SIZe <NR3>
TRIGger:A:BUS:I2C:DATa:SIZe?
```

Related Commands

TRIGger:A:BUS:I2C:DATa:DIRection, TRIGger:A:BUS:I2C:DATa:FORMat, TRIGger:A:BUS:I2C:DATa:VALue

Arguments

`<NR3>` specifies the data size in bytes.

Examples

`TRIGGER:A:BUS:I2C:DATA:SIZE 3` sets the I2C bus trigger data size to one byte.

`TRIGGER:A:BUS:I2C:DATA:SIZE?` might return `:TRIGGER:A:BUS:I2C:DATA:SIZE 1`, indicating that the data size for the I2C bus trigger is set to one byte.

TRIGger:A:BUS:I2C:DATa:VALue

NOTE. *Not available on some models.*

This command sets or queries the data value of the data token for an I2C trigger when the trigger condition is Data or Addr + Data.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:I2C:DATa:VALue <string>
TRIGger:A:BUS:I2C:DATa:VALue?
```

Arguments <string> specifies the data value. The valid characters are 0, 1, or X representing a binary number.

Examples TRIGGER:A:BUS:I2C:DATA:VALUE"00001111" sets the I2C bus trigger data value to "00001111".

TRIGGER:A:BUS:I2C:DATA:VALUE? might return :TRIGGER:A:BUS:I2C:DATA:VALUE "XXXXXX01", indicating that the data value for the I2C bus trigger is set to "XXXXXX01".

TRIGger:A:BUS:LIN:CONDition

This command sets or queries the LIN bus trigger condition.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:LIN:CONDition
{DATA|IDANDDATA|ERRor|IDentifier|SLEEP|SYNC|WAKEup}
TRIGger:A:BUS:LIN:CONDition?
```

Arguments Arguments are the available trigger conditions.

Examples TRIGger:A:BUS:LIN:CONDition DATA sets the trigger condition to DATA.

TRIGger:A:BUS:LIN:CONDition? might return :TRIGGER:A:BUS:LIN:CONDITION SYNC, indicating the trigger condition is SYNC.

TRIGger:A:BUS:LIN:DATa:FORMat

This command sets or queries the LIN bus trigger data format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:LIN:DATa:FORMat {BINary|HEX}
TRIGger:A:BUS:LIN:DATa:FORMat?
```

Arguments `BINary` specifies the data format as binary.

`HEX` specifies the data format as hexadecimal.

Examples `TRIGger:A:BUS:LIN:DATa:FORMat HEX` sets the data format to hexadecimal.

`TRIGger:A:BUS:LIN:DATa:FORMat?` might return `:TRIGGER:A:BUS:LIN:DATA:FORMAT BINARY`, indicating the data format is binary.

TRIGger:A:BUS:LIN:DATa:QUALifier

This command sets or queries the LIN bus trigger data qualifier.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:LIN:DATa:QUALifier
{EQUal|LESSEQual|MOREEQual|UNEQual|LESSthan|MOREthan}
TRIGger:A:BUS:LIN:DATa:QUALifier?
```

Arguments Arguments specify the available data qualifiers.

Examples `TRIGger:A:BUS:LIN:DATa:QUALifier LESSTHAN` sets the data qualifier to less than the data value.

`TRIGger:A:BUS:LIN:DATa:QUALifier?` might return `:TRIGGER:A:BUS:LIN:DATA:QUALIFIER EQUAL`, indicating the data qualifier is set to equal the data value.

TRIGger:A:BUS:LIN:DATa:SIZe

This command sets or queries the LIN bus trigger data size (the number of contiguous CAN data bytes to trigger on).

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:LIN:DATa:SIZe <nr3>
TRIGger:A:BUS:LIN:DATa:SIZe?
```

Arguments <nr3> specifies the data size.

Examples `TRIGger:A:BUS:LIN:DATa:SIZe 2` sets the data size to 2.

`TRIGger:A:BUS:LIN:DATa:SIZe?` might return `:TRIGGER:A:BUS:LIN:DATA:SIZE 1`, indicating the data size is 1.

TRIGger:A:BUS:LIN:DATa:VALue

This command sets or queries the LIN bus trigger data value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:LIN:DATa:VALue <string>
TRIGger:A:BUS:LIN:DATa:VALue?
```

Arguments <string> specifies the data value.

Examples `TRIGger:A:BUS:LIN:DATa:VALue 1010` sets the data value to XXXX1010.

`TRIGger:A:BUS:LIN:DATa:VALue?` might return `:TRIGGER:A:BUS:LIN:DATA:VALUE "XXX10101"`, indicating the data value is set to XXX10101.

TRIGger:A:BUS:LIN:ERRTYPE

This command sets or queries the LIN bus trigger error type.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:LIN:ERRTYPE {CHecksum|PARity|SYNC}
TRIGger:A:BUS:LIN:ERRTYPE?
```

Arguments

`CHecksum` specifies the error type is checksum.

`PARity` specifies the error type is parity.

`SYNC` specifies the error type is sync.

Examples

`TRIGger:A:BUS:LIN:ERRTYPE CHECKsum` sets the error type to checksum.

`TRIGger:A:BUS:LIN:ERRTYPE?` might return `:TRIGGER:A:BUS:LIN:ERRTYPE SYNC`, indicating the error type is sync.

TRIGger:A:BUS:LIN:IDentifier:FORMat

This command sets or queries the LIN bus trigger identifier format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:LIN:IDentifier:FORMat {BINary|HEX}
TRIGger:A:BUS:LIN:IDentifier:FORMat?
```

Arguments

`BINary` specifies the identifier format as binary.

`HEX` specifies the identifier format as hexadecimal.

Examples

`TRIGger:A:BUS:LIN:IDentifier:FORMat HEX` sets the identifier format to hexadecimal.

TRIGger:A:BUS:LIN:IDentifier:FORMat? might return :TRIGGER:A:BUS:LIN:IDENTIFIER:FORMAT BINARY, indicating the identifier format is binary.

TRIGger:A:BUS:LIN:IDentifier:VALue

This command sets or queries the LIN trigger identifier value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:LIN:IDentifier:VALue <string>
TRIGger:A:BUS:LIN:IDentifier:VALue?
```

Arguments <string> specifies the identifier value.

Examples TRIGger:A:BUS:LIN:IDentifier:VALue "1001" sets the identifier value to XX1001.

TRIGger:A:BUS:LIN:IDentifier:VALue? might return :TRIGGER:A:BUS:LIN:IDENTIFIER:VALUE "X11001", indicating the identifier value is X1101.

TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat

This command sets the data format of the address (either binary or hex) to use in a MIL-STD-1553 bus command word trigger. The trigger condition needs to be set to COMMAND.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax
```
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat {BINary|HEX}
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples

TRIGGER:A:BUS:MIL1553B:COMMAND:ADDRESS:FORMAT HEX sets the data format to HEX.

TRIGGER:A:BUS:MIL1553B:COMMAND:ADDRESS:FORMAT? might return BINARY, indicating that the data format is set to binary.

TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier

This command specifies the qualifier to use when triggering on a MIL-STD-1553 bus command word remote terminal address field. The trigger condition needs to be set to COMMAND.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier
{LESSthan|MOREthan|EQual|UNEQual|LESSEQual|MOREEQual}
```

```
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue

Arguments

LESSthan sets the Command Address qualifier to less than.

MOREthan sets the Command Address qualifier to greater than.

EQual sets the Command Address qualifier to equal.

UNEQual sets the Command Address qualifier to not equal.

LESSEQual sets the Command Address qualifier to less than or equal.

MOREEQual sets the Command Address qualifier to greater than or equal.

Examples

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:ADDRESS:QUALIFIER EQual sets the qualifier to be used with the remote terminal address to equal.

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:ADDRESS:QUALIFIER? might return EQUAL.

TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue

This command specifies the value of the 5-bit remote terminal address to use in a MIL-STD-1553 bus command word trigger. The trigger condition needs to be set to COMMAND.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue <QString>
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier

Arguments QString is a quoted string of up to 5 characters, where the allowable characters are 0, 1, and X. The default is all X's (don't care). The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples TRIGGER:A:BUS:B1:MIL1553B:COMMAND:ADDRESS:VALUE "01000" sets the remote terminal address to be used in the trigger to 01000.

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:ADDRESS:VALUE? might return 01000.

TRIGger:A:BUS:MIL1553B:COMMAND:COUNt

This command specifies the bit pattern for the 5-bit Word Count/Mode Code sub-address field to use in a MIL-STD-1553 bus command word trigger. (Use the command TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress to specify Word Count or Mode Code.) In Word Count mode, this field defines the number of data words to be transmitted or received, depending on the T/R bit setting. (Use the command TRIGger:A:BUS:MIL1553B:COMMAND:TRBit to set the T/R bit.) A word count value of 0 actually indicates a transfer of 32 data words. The trigger condition needs to be set to COMMAND.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt <QString>
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress

TRIGger:A:BUS:MIL1553B:COMMAND:TRBit

Arguments

QString is a quoted string of up to 5 characters, where the allowable characters are 0, 1 and X.

Examples

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:COUNT "01000" sets the bit pattern for the Word Count/Mode Code field to 01000.

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:COUNT? might return "XXXXX".

TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat

This command specifies the data format (binary or hex) of the bit pattern for the 5-bit Word Count/Mode Code field to use in a MIL-STD-1553 bus command word trigger. The trigger condition needs to be set to COMMAND.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat {BINary|HEX}
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples

TRIGGER:A:BUS:MIL1553B:COMMAND:COUNT:FORMAT sets the data format to HEX.

TRIGGER:A:BUS:MIL1553B:COMMAND:COUNT:FORMAT? might return BINARY, indicating that the data format is set to binary.

TRIGger:A:BUS:MIL1553B:COMMAND:PARity

This command specifies the parity to use in a MIL-STD-1553 bus command word trigger. The trigger condition needs to be set to COMMAND.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:COMMAND:PARity
{0|1|X|ZERo|ONE|NOCARE|OFF|ON|DONTCare}
```

```
TRIGger:A:BUS:MIL1553B:COMMAND:PARity?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0

1

X sets the value to X (“don’t care”), which is the default.

ZERO sets the value to 0.

ONE sets the value to 1.

NOCARE sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples TRIGGER:A:BUS:B1:MIL1553B:COMMAND:PARITY ZERo sets the Command word parity bit to 0.

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:PARITY? might return 0, 1 or X.

TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress

This command specifies the 5 bit sub-address to use in a MIL-STD-1553 bus command word sub-address trigger. When the sub-address value is set to 00000 or 11111 binary, it specifies that the command is a “Mode Code” command. Any other value specifies that it is a “Word Count” command. The default is all X’s (don’t care). The trigger condition needs to be set to COMMAND.

NOTE. *The sub-address field is used to direct data to different functions within the subsystem for values 1 - 30.*

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax
```
TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress <QString>
TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:COMMAND:COUNt

Arguments QString is a quoted string of up to 5 characters, where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `TRIGGER:A:BUS:B1:MIL1553B:COMMAND:SUBADDRESS "01000"` sets the trigger sub-address to 01000, which means that it is a "Word Count" command.

TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:FORMat

This command specifies the data format (binary or hex) of the sub-address field bit pattern to use in a MIL-STD-1553 bus command word sub-address trigger. The trigger condition needs to be set to `COMMAND`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax
```
TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:FORMat
{BINary|HEX}
TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:FORMat?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples

TRIGGER:A:BUS:MIL1553B:COMMAND:SUBADDRESS:FORMAT HEX sets the data format to HEX.

TRIGGER:A:BUS:MIL1553B:COMMAND:SUBADDRESS:FORMAT? might return BINARY, indicating that the data format is set to binary.

TRIGger:A:BUS:MIL1553B:COMMAND:TRBit

This command specifies a trigger on either a “transmit” or a “receive” signal for the transmit/receive bit (bit 9) for a MIL-STD-1553 bus command word trigger. The trigger condition needs to be set to COMMAND.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:COMMAND:TRBit {RX|TX|X|DONTCare}
TRIGger:A:BUS:MIL1553B:COMMAND:TRBit?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:COMMAND:COUNt

Arguments

TX (logic 1) directs the instrument to trigger on a TX or "transmit" from a remote terminal .

RX (logic 0) directs the instrument to trigger on an RX or "receive" from a remote terminal.

X, DONTCare indicates “don’t care".

Examples

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:TRBIT TX directs the instrument to trigger on a TX or “transmit” signal from a remote terminal.

TRIGGER:A:BUS:B1:MIL1553B:COMMAND:TRBIT TX? might return TX.

TRIGger:A:BUS:MIL1553B:CONDition

This command specifies which word type or condition within a MIL-STD-1553 bus word to trigger on.

NOTE. *There are three types of MIL-STD-1553 (Aircraft) bus words: Command, Status, and Data, all sharing a common structure twenty bits in length. The first three bits are used as a synchronization field. The last bit is the odd parity of the previous 16 bits.*

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:CONDition
{SYNC|COMMAND|STATUS|DATA|TIMe|ERRor}

TRIGger:A:BUS:MIL1553B:CONDition?
```

Related Commands Most of the other `TRIGger:A:BUS:B<X>:MIL1553B` commands are impacted by the setting of this command.

BUS:B<x>:MIL1553B:POLarity

BUS:B<x>:MIL1553B:SOUrce

Arguments

`SYNC` refers to the 3-bit sync pulse that precedes each word.

`COMMAND` is one of 3 16-bit word types.

`STATUS` is one of 3 16-bit word types.

`DATA` is one of 3 16-bit word types.

`TIMe` specifies a trigger on either the RT (remote terminal response time), or the IMG (Inter-message Gap). Use the commands TRIGger:A:BUS:MIL1553B:TIME:QUALifier, TRIGger:A:BUS:MIL1553B:TIME:LESSLimit, and TRIGger:A:BUS:MIL1553B:TIME:MORELimit to specify the time parameters.

`ERRor` specifies to trigger upon a signaling error. (You can specify which type of error - Parity, Sync, Manchester or Non-contiguous Data - by using the TRIGger:A:BUS:MIL1553B:ERRTYPE command.)

NOTE.

Use the `TRIGger:A:BUS:B<x>:MIL1553B:CONDition:STATUS:BIT` *commands to set the following bits:*

9 – message error
TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME

10 – instrumentation
TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR

11 – SRQ (service request)
TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ

15 – BCR (broadcast command received)
TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR

16 – busy
TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY

17 – subsystem flag
TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF

18 – DBCA (dynamic bus control acceptance)
TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA

19 – terminal flag
TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF

Examples

`TRIGGER:A:BUS:B1:MIL1553B:CONDITION STATUS` specifies to trigger on the word type `STATUS`.

`TRIGGER:A:BUS:B1:MIL1553B:CONDITION?` might return `DATA`.

TRIGger:A:BUS:MIL1553B:DATA:FORMat

This command specifies the format (binary or hex) of the bit pattern to use for a MIL-STD-1553 bus data word trigger. The trigger condition needs to be set to `DATA`.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:DATA:FORMat {BINary|HEX}
TRIGger:A:BUS:MIL1553B:DATA:FORMat?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition, TRIGger:A:BUS:MIL1553B:DATA:PARity, TRIGger:A:BUS:MIL1553B:DATA:VALue

Arguments

BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples

TRIGGER:A:BUS:MIL1553B:DATA:FORMAT sets the data format to HEX.

TRIGGER:A:BUS:MIL1553B:DATA:FORMAT? might return BINARY, indicating that the data format is set to binary.

TRIGger:A:BUS:MIL1553B:DATA:PARity

This command specifies the parity to use when triggering on a MIL-STD-1553 bus data word parity bit. The trigger condition must be set to DATA.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:DATA:PARity
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:DATA:PARity?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:DATA:VALue

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

TRIGGER:A:BUS:MIL1553B:DATA:PARITY 1 sets the data parity bit to be used in a MIL-STD-1553 bus trigger to 1.

TRIGGER:A:BUS:MIL1553B:DATA:PARITY? might return 0, indicating that the data parity bit to be used in a MIL-STD-1553 bus trigger is set to 0.

TRIGger:A:BUS:MIL1553B:DATA:VALue

This command specifies the value to use for a MIL-STD-1553 bus data word trigger. This is a 16-bit field. The default is all X's (don't care). The trigger condition must be set to DATA.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:DATA:VALue <QString>
TRIGger:A:BUS:MIL1553B:DATA:VALue?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:DATA:PARity

Arguments QString is a quoted string of up to 16 characters, where the allowable characters are 0, 1, and X. The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples TRIGGER:A:BUS:B1:MIL1553B:DATA:VALUE "01000" sets the data binary pattern to be used in a MIL-STD-1553 bus trigger to 01000.

TRIGGER:A:BUS:B1:MIL1553B:DATA:VALUE? might return XXXXXXXXXXX01000.

TRIGger:A:BUS:MIL1553B:ERRTYPE

This command specifies the signaling error type to use for a MIL-STD-1553 bus data word trigger: Parity, Sync, Manchester, or Data. The trigger condition must be set to ERRor.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:ERRTYPE {PARity|SYNC|MANCHester|DATA}
TRIGger:A:BUS:MIL1553B:ERRTYPE?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments PARity – a failed parity check.

SYNC – the high to low, or low to high transition doesn't happen in the middle of the sync time as it should.

MANCHester – no transition in a bit time.

DATA – a non-contiguous data error.

Examples TRIGGER:A:BUS:B1:MIL1553B:ERRTYPE PARITY sets the error type to be used in the trigger to PARITY.

TRIGger:A:BUS:B1:MIL1553B:ERRTYPE? might return SYNC.

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:FORMat

This command specifies the format (binary or hex) of the bit pattern to use when triggering on the MIL-STD-1553 status address field. The trigger condition needs to be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax
```
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:FORMat {BINary|HEX}
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:FORMat?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue

Arguments BINary specifies the data format as binary.

HEX specifies the data format as hexadecimal.

Examples TRIGGER:A:BUS:MIL1553B:STATUS:ADDRESS:FORMAT HEX sets the data format to HEX.

TRIGGER:A:BUS:MIL1553B:STATUS:ADDRESS:FORMAT? might return BINARY, indicating that the data format is set to binary.

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier

This command specifies the qualifier to use when triggering on the MIL-STD-1553 status address field. The default is EQUAL. The trigger condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier
{LESSthan|MOREthan|EQual|UNEQual|LESSEQual|MOREEQual}
```

```
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue

Arguments LESSthan sets the Status Address qualifier to less than.

MOREthan sets the Status Address qualifier to greater than.

EQual sets the Status Address qualifier to equal.

UNEQual sets the Status Address qualifier to not equal.

LESSEQual sets the Status Address qualifier to less than or equal.

MOREEQual sets the Status Address qualifier to greater than or equal.

Examples TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:QUALIFIER MORETHAN sets the qualifier to be used with the address field to greater than.

TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:QUALIFIER? might return MORETHAN.

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue

This command specifies the value of the 5-bit remote terminal address to use when triggering on the MIL-STD-1553 status address field. The default is all X's (don't care). The trigger condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue <QString>
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier

Arguments QString is a quoted string of up to 5 characters, where the allowable characters are 0, 1, and X (don't care). The bits specified in the quoted string replace the least significant bits, leaving any unspecified upper bits unchanged.

Examples `TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:VALUE "01000"` sets the value of the 5-bit remote terminal address to be used in the trigger, when the qualifier has been set to `LESSthan, MOREthan, EQual, UNEQual, LESSEQual` or `MOREEQual`.

`TRIGGER:A:BUS:B1:MIL1553B:STATUS:ADDRESS:VALUE?` might return "01000".

TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR

This command specifies the status word broadcast command received (BCR) bit value (bit 15) to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to `STATUS`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

`0, ZERO` sets the value to 0.

`1, ONE` sets the value to 1.

`X, NOCARE, DONTCare` sets the value to X ("don't care"), which is the default.

`OFF` sets the value to 0.

`ON` sets the value to 1.

Examples

`TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:BCR NOCARE` sets the BCR trigger bit value to X.

`TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:BCR?` might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY

This command specifies the status word busy bit value (bit 16) to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to `STATUS`.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

Arguments

`0, ZERO` sets the value to 0.

`1, ONE` sets the value to 1.

`X, NOCARE, DONTCare` sets the value to X ("don't care"), which is the default.

`OFF` sets the value to 0.

`ON` sets the value to 1.

Examples

`TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:BUSY NOCARE` sets the status word busy bit trigger value to X.

`TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:BUSY?` might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA

This command specifies the status word dynamic bus control acceptance (DBCA) bit value (bit 18) to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}
```

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X ("don't care"), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:DBCA NOCARE sets the status word dynamic bus control acceptance (DBCA) bit trigger value to X.

TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:DBCA? might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR

This command specifies the status word instrumentation bit value (bit 10) to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

`0, ZERO` sets the value to 0.

`1, ONE` sets the value to 1.

`X, NOCARE, DONTCare` sets the value to X ("don't care"), which is the default.

`OFF` sets the value to 0.

`ON` sets the value to 1.

Examples

`TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:INSTR ZERo` sets the status word instrumentation bit trigger value to 0.

`TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:INSTR?` might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME

This command specifies the status word message error bit value (bit 9) to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to `STATUS`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

`0, ZERO` sets the value to 0.

`1, ONE` sets the value to 1.

`X, NOCARE, DONTCare` sets the value to X ("don't care"), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:ME NOCARE sets the status word message error bit trigger value to X.

TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:ME? might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity

This command specifies the parity to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to STATUS.

Group Trigger

Syntax
```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity
{0|1|X|ZERo|ONE|NOCARE|DONTCare|ON|OFF}
TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity?
```

Arguments 0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X ("don't care"), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:PARITY NOCARE sets the trigger parity to X.

TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:PARITY? might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ

This command specifies the status word service request (SRQ) bit value (bit 11) to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to STATUS.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

`0, ZERO` sets the value to 0.

`1, ONE` sets the value to 1.

`X, NOCARE, DONTCare` sets the value to X ("don't care"), which is the default.

`OFF` sets the value to 0.

`ON` sets the value to 1.

Examples

`TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:SRQ NOCARE` sets the status word SRQ bit trigger value to X.

`TRIGGER:A:BUS:B1:MIL1553B:STATUS:BIT:SRQ?` might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF

This command specifies the status word subsystem flag bit value (bit 17) to use when triggering on the MIL-STD-1553 status bit field. The default is all X's (don't care). The trigger condition must be set to `STATUS`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:SUBSF NOCARE sets the status word subsystem flag bit trigger value to X.

TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:SUBSF? might return 1.

TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF

This command specifies the status word terminal flag bit value (bit 19) to use when triggering on the MIL-STD-1553 status bit field. The default is all X’s (don’t care). The trigger condition must be set to STATUS.

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF
{0|1|X|ZERo|ONE|NOCARE|DONTCare|OFF|ON}

TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

Arguments

0, ZERO sets the value to 0.

1, ONE sets the value to 1.

X, NOCARE, DONTCare sets the value to X (“don’t care”), which is the default.

OFF sets the value to 0.

ON sets the value to 1.

Examples

TRIGGER:A:BUS:B1:MIL1553B:STAT:BIT:TF NOCARE sets the status word terminal flag bit trigger value to X.

TRIG:A:BUS:B1:MIL1553B:STAT:BIT:TF? might return 1.

TRIGger:A:BUS:MIL1553B:TIME:LESSLimit

This command specifies either the minimum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit or the minimum inter-message gap (IMG). The instrument will trigger when it finds an RT or IMG that meets the specified conditions. The trigger condition must be set to `TIMe`.

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:TIME:LESSLimit <NR3>
TRIGger:A:BUS:MIL1553B:TIME:LESSLimit?
```

Related Commands TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:TIME:MORELimit

Arguments `<NR3>` is a floating point number that specifies either the minimum remote terminal response time (RT) or the inter-message gap (IMG) in seconds.

Examples `TRIGGER:A:BUS:B1:MIL1553B:TIME:LESSLIMIT 3.0000E-6` would set either the minimum remote terminal response time (RT) or the minimum inter-message gap (IMG) to 3 microseconds.

`TRIGGER:A:BUS:B1:MIL1553B:TIME:LESSLIMIT?` might return 2.0000e-6, indicating that the minimum RT or IMG has been set to 2 microseconds.

TRIGger:A:BUS:MIL1553B:TIME:MORELimit

This command specifies either the maximum remote terminal response time (RT) limit, in seconds, for the amount of time the terminal has to transmit or the maximum inter-message gap (IMG). The instrument will trigger when it finds an RT or IMG that meets the specified conditions. The trigger condition must be set to `TIMe`,

Conditions Requires the SR-AERO Serial Triggering and Analysis Application.

Group Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:TIME:MORELimit <NR3>
TRIGger:A:BUS:MIL1553B:TIME:MORELimit?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:TIME:LESSLimit

Arguments

<NR3> is a floating point number that specifies either the maximum remote terminal response time (RT) or the maximum inter-message gap (IMG) in seconds.

Examples

TRIGGER:A:BUS:B1:MIL1553B:TIME:MORELIMIT 80.0000E-6 would set either the maximum remote terminal response time (RT) or the inter-message gap (IMG) to be used in the trigger to 80 microseconds.

TRIGGER:A:BUS:B1:MIL1553B:TIME:MORELIMIT? might return 100.0000e-6, indicating that the maximum RT or IMG has been set to 100 microseconds.

TRIGger:A:BUS:MIL1553B:TIME:QUALifier

This command specifies the qualifier to use when triggering on either the remote terminal response time (RT) or the inter-message gap (IMG). The instrument will trigger when it finds an RT or IMG that meets the specified conditions. The trigger condition must be set to TIMe,

Conditions

Requires the SR-AERO Serial Triggering and Analysis Application.

Group

Trigger

Syntax

```
TRIGger:A:BUS:MIL1553B:TIME:QUALifier
{LESSthan|MOREthan|INrange|OUTrange}

TRIGger:A:BUS:MIL1553B:TIME:QUALifier?
```

Related Commands

TRIGger:A:BUS:MIL1553B:CONDition

TRIGger:A:BUS:MIL1553B:TIME:LESSLimit

TRIGger:A:BUS:MIL1553B:TIME:MORELimit

Arguments

LESSthan sets the time qualifier to less than minimum.

MOREthan sets the time qualifier to greater than maximum

INrange sets the time qualifier to inside range.

OUTrange sets the time qualifier to out of range.

Examples

TRIGGER:A:BUS:B1:MIL1553B:TIME:QUALIFIER LESSthan sets the time qualifier to less than minimum.

TRIGGER:A:BUS:B1:MIL1553B:TIME:QUALIFIER? might return MORETHAN.

TRIGger:A:BUS:PCIE:CHARacter:CHAR

This command sets or queries the PCIE bus trigger character.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:PCIE:CHARacter:CHAR <string>
TRIGger:A:BUS:PCIE:CHARacter:CHAR?
```

Related Commands

TRIGger:A:BUS:PCIE:PATtern:ORDERedset, TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, TRIGger:A:BUS:PCIE:CHARacter:CHAR, TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

Arguments

<string> specifies the trigger character.

Examples

TRIGger:A:BUS:PCIE:CHARacter:CHAR “D10.2” sets the trigger character to D10.2.

TRIGger:A:BUS:PCIE:CHARacter:CHAR? might return :TRIGGER:A:BUS:PCIE:CHARACTER:CHAR "K28.5", indicating the trigger character is K28.5.

TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

This command sets or queries the PCIE bus trigger character symbol minus (negative disparity).

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus <string>
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus?
```

Related Commands TRIGger:A:BUS:PCIE:PATtern:ORDERedset, TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, TRIGger:A:BUS:PCIE:CHARacter:CHAR, TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

Arguments `<string>` specifies the minus character symbol.

Examples `TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus` "10000010" sets the minus character symbol to 10000010.

`TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus?` might return `:TRIGGER:A:BUS:PCIE:CHARACTER:SYMBOL:MINUS "0011111010"`, indicating the minus character symbol is 0011111010.

TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS

This command sets or queries PCIE bus trigger character symbol plus (positive disparity).

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS <string>
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS?
```

Related Commands TRIGger:A:BUS:PCIE:PATtern:ORDERedset, TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, TRIGger:A:BUS:PCIE:CHARacter:CHAR, TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

Arguments `<string>` specifies the character symbol plus.

Examples `TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS “10110010”` sets the character symbol plus to 10110010.

`TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS?` might return `:TRIGGER:A:BUS:PCIE:CHARACTER:SYMBOL:PLUS "1100000101"`, indicates that the character symbol plus is 1100000101.

TRIGger:A:BUS:PCIE:CONDition

This command sets or queries the PCIE bus trigger condition.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:PCIE:CONDition
{ANYControl|CHARacter|ERROR|PATtern}
TRIGger:A:BUS:PCIE:CONDition?
```

Arguments Arguments are the available trigger conditions.

Examples `TRIGger:A:BUS:PCIE:CONDition ERROR` sets the trigger condition to error.

`TRIGger:A:BUS:PCIE:CONDition?` might return `:TRIGGER:A:BUS:PCIE:CONDITION PATTERN`, indicating the trigger condition is pattern.

TRIGger:A:BUS:PCIE:DISParity

This command sets or queries the PCIE bus trigger disparity.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:PCIE:DISParity {EITher|NEGAtive|POSITIVe}
TRIGger:A:BUS:PCIE:DISParity?
```

Arguments `NEGative` specifies negative disparity.

`POSITIVe` specifies positive disparity.

EITher specifies either disparity.

Examples

TRIGger:A:BUS:PCIE:DISParity NEGATIVE sets the disparity to negative.

TRIGger:A:BUS:PCIE:DISParity? might return :TRIGGER:A:BUS:PCIE:DISPARITY EITHER, indicates that the disparity to either.

TRIGger:A:BUS:PCIE:ERROR

This command sets or queries the PCIE bus trigger error.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:PCIE:ERROR {CHARacter|DISParity}
TRIGger:A:BUS:PCIE:ERROR?
```

Arguments

CHARACTER sets the instrument to trigger on a character error.

DISPARITY sets the instrument to trigger on a disparity error.

Examples

TRIGger:A:BUS:PCIE:ERROR Disparity sets the instrument to trigger on a disparity error.

TRIGger:A:BUS:PCIE:ERROR? might return :TRIGGER:A:BUS:PCIE:ERROR CHARACTER, indicating the instrument is set to trigger on a character error..

TRIGger:A:BUS:PCIE:FORMat

This command sets or queries the PCIe bus trigger format.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:PCIE:FORMat {ORDERedset|SYMbol|CHAR}
TRIGger:A:BUS:PCIE:FORMat?
```

Arguments Arguments are the available trigger formats.

Examples `TRIGger:A:BUS:PCIE:FORMat Symbol` sets the trigger format to symbol.

`TRIGger:A:BUS:PCIE:FORMat?` might return `:TRIGGER:A:BUS:PCIE:FORMAT ORDEREDSET`, indicating the trigger format is set to format.

TRIGger:A:BUS:PCIE:PATtern:CHAR<x>

This command sets or queries the PCIE bus trigger pattern character. The x specifies the character and can be 1 through 4.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:PCIE:PATtern:CHAR<x> <string>
TRIGger:A:BUS:PCIE:PATtern:CHAR<x>?
```

Related Commands TRIGger:A:BUS:PCIE:PATtern:ORDERedset, TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, TRIGger:A:BUS:PCIE:CHARacter:CHAR, TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

Arguments `<string>` specifies the trigger pattern character. Valid characters are any valid PCIe/8b10b character (KCodes, DCodes, and KCode mnemonics like COM and SKP) or X, the don't care character.

Examples `TRIGger:A:BUS:PCIE:PATtern:CHAR3 "COM"` sets the trigger patter character to COM, or K28.5.

`TRIGger:A:BUS:PCIE:PATtern:CHAR1?` might return `:TRIGGER:A:BUS:PCIE:PATTERN:CHAR1 "END"`, indicating the trigger pattern character is END, or K29.7.

`TRIGger:A:BUS:PCIE:PATtern:CHAR2?` might return `:TRIGGER:A:BUS:PCIE:PATTERN:CHAR2 "X"`, indicating the trigger pattern character is X, or don't care.

TRIGger:A:BUS:PCIE:PATtern:ORDERedset

This command sets or queries the PCIE bus trigger pattern ordered set.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:PCIE:PATtern:ORDERedset {EI|FTS|SKP}
TRIGger:A:BUS:PCIE:PATtern:ORDERedset?
```

Related Commands TRIGger:A:BUS:PCIE:PATtern:ORDERedset, TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, TRIGger:A:BUS:PCIE:CHARacter:CHAR, TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

Arguments Arguments are the available ordered sets.

`INVALID` is an additional argument that can be returned by a query.

Examples `TRIGger:A:BUS:PCIE:PATtern:ORDERedset SKP` sets the trigger pattern ordered set to SKP.

`TRIGger:A:BUS:PCIE:PATtern:ORDERedset?` might return `:TRIGGER:A:BUS:PCIE:PATTERN:ORDEREDSET SKP`, indicates the ordered set is set to SKP.

TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>

This command sets or queries the PCIE bus trigger pattern symbol minus (negative disparity). The x specifies the minus character and can be 1 through 4.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x> <string>
TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>?
```

Related Commands TRIGger:A:BUS:PCIE:PATtern:ORDERedset, TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, TRIGger:A:BUS:PCIE:CHARacter:CHAR, TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

Arguments <string> specifies the trigger pattern minus symbol.

Examples TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus1 “1010101011” sets the trigger pattern symbol minus to 1010101011.

TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus1? might return :TRIGGER:A:BUS:PCIE:PATTERN:SYMBOL:MINUS1 "0011111010", indicating the plus symbol is 0011111010.

TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>

This command sets or queries the PCIE bus trigger pattern plus symbol (positive disparity). The x specifies the plus character and can be 1 through 4.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x> <string>
TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>?
```

Related Commands TRIGger:A:BUS:PCIE:PATtern:ORDERedset, TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, TRIGger:A:BUS:PCIE:CHARacter:CHAR, TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus

Arguments <string> specifies the pattern symbol plus.

Examples TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS1 “1010101011” sets the trigger pattern symbol plus to 1010101011.

TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS1? might return :TRIGGER:A:BUS:PCIE:PATTERN:SYMBOL:PLUS1 "1100000101", indicating the plus symbol is 1100000101.

TRIGger:A:BUS:RS232C:CONDition

NOTE. *Not available on some models.*

This command sets or queries the condition for an RS-232C bus trigger.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:RS232C:CONDition {DATa|EOp|PARItyerror|STARt}
TRIGger:A:BUS:RS232C:CONDition?
```

Arguments

`DATa` sets the Trigger on condition to Data.

`EOp` sets the Trigger on condition to End of Packet.

`PARItyerror` sets the Trigger on condition to Parity Error.

`STARt` sets the Trigger on condition to Start.

Examples

`TRIGGER:A:BUS:RS232C:CONDITION DATA` sets the RS-232C trigger on condition to Data.

`TRIGGER:A:BUS:RS232C:CONDITION?` might return `:TRIGGER:A:BUS:RS232C:CONDITION START`, indicating that the RS-232C trigger on condition is set to Start.

TRIGger:A:BUS:RS232C:DATa:FORMat

NOTE. *Not available on some models.*

This command sets or queries the data format for the RS232C bus trigger.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:RS232C:DATa:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:RS232C:DATa:FORMat?
```

Arguments

`BINary` specifies that the data format for the RS232C bus trigger is Binary.

`HEXadecimal` specifies that the data format for the RS232C bus trigger is Hexadecimal.

Examples `TRIGGER:A:BUS:RS232C:DATA:FORMAT BINARY` sets the data format to BINary.

`TRIGGER:A:BUS:RS232C:DATA:FORMAT?` might return `:TRIGGER:A:BUS:RS232C:DATA:FORMAT BINARY`, indicating that the data format for the RS232C bus trigger is set to BINary.

TRIGger:A:BUS:RS232C:DATa:SIZe

NOTE. *Not available on some models.*

This command sets or queries the length of the data string in bytes to be used for an RS-232C trigger when the trigger condition is Data.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:RS232C:DATa:SIZe <NR3>
TRIGger:A:BUS:RS232C:DATa:SIZe?
```

Arguments `<NR3>` specifies the data size in bytes.

Examples `TRIGGER:A:BUS:RS232C:DATA:SIZE 3` sets three bytes data size for the RS-232C bus trigger.

`TRIGGER:A:BUS:RS232C:DATA:SIZE?` might return `:TRIGGER:A:BUS:RS232C:DATA:SIZE 2`, indicating that the data size for the RS-232C bus trigger is set to two bytes.

TRIGger:A:BUS:RS232C:DATa:VALue

NOTE. *Not available on some models.*

This command sets or queries the data address string used for the RS-232 bus trigger when the trigger condition is set to Data.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:RS232C:DATa:VALue <string>
TRIGger:A:BUS:RS232C:DATa:VALue?
```

Arguments <string> specifies the address value. The argument is a string of 0, 1, or X representing a binary number.

Examples TRIGGER:A:BUS:RS232C:DATA:VALUE "011XX11" sets the data address string used for the RS-232 bus trigger to 011XX11.

TRIGGER:A:BUS:RS232C:DATA:VALUE? might return :TRIGGER:A:BUS:RS232C:DATA:VALUE "XXXXXX01", indicating that the data address string used for the RS-232 bus trigger is set to "XXXXXX01"

TRIGger:A:BUS:S8B10B:CHARacter:CHAR

Sets or queries the 8B10B character when the Trigger Condition is set to Character.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:BUS:S8B10B:CHARacter:CHAR <string>
TRIGger:A:BUS:S8B10B:CHARacter:CHAR?
```

Related Commands TRIGger:A:BUS:S8B10B:CONDition, TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus, TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS

Arguments <string> is the 8B10B character.

Examples TRIGGER:A:BUS:S8B10B:CHARACTER:CHAR"K28.5" sets the character to K28.5.

TRIGGER:A:BUS:S8B10B:CHARACTER:CHAR? might return :TRIGGER:A:BUS:S8B10B:CHARACTER:CHAR "K28.3", indicating that the character is set to K28.3.

TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus

Sets or queries the 8B10B character bit pattern (negative disparity) when the Trigger Condition is set to Character.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus <string>
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus?
```

Related Commands TRIGger:A:BUS:S8B10B:CONDition, TRIGger:A:BUS:S8B10B:CHARacter:CHAR, TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS

Arguments <string> is the 8B10B character bit pattern.

Examples TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:MINUS "0011111010" sets the character bit pattern to 0011111010.

TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:MINUS? might return :TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:MINUS "0011110011", indicating that the character bit pattern is set to 0011110011.

TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS

Sets or queries the Character bit pattern (positive disparity view) when the Trigger Condition is set to Character.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS <string>
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS?
```

Related Commands TRIGger:A:BUS:S8B10B:CONDition, TRIGger:A:BUS:S8B10B:CHARacter:CHAR, TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus

Arguments <string> is the 8B10B character bit pattern.

Examples TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:PLUS "1100000101" sets the character bit pattern to 1100000101.

TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:PLUS? might return :TRIGGER:A:BUS:S8B10B:CHARACTER:SYMBOL:PLUS "1100001100", indicating that the character bit pattern is 1100001100.

TRIGger:A:BUS:S8B10B:CONDition

Sets or queries the trigger condition for the 8B10B bus.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:BUS:S8B10B:CONDition
{ANYControl|CHARacter|ERROR|PATtern}
TRIGger:A:BUS:S8B10B:CONDition?
```

Related Commands TRIGger:A:BUS:S8B10B:CHARacter:CHAR, TRIGger:A:BUS:S8B10B:ERROR, TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus, TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS, TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>, TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>

Arguments

`ANYControl`

`CHARacter`

`ERROR`

`PATtern`

Examples

`TRIGGER:A:BUS:S8B10B:CONDITION` ERROR sets the oscilloscope to trigger on an error.

`TRIGGER:A:BUS:S8B10B:CONDITION?` might return`:TRIGGER:A:BUS:S8B10B:CONDITION PATtern`, indicating that the oscilloscope will trigger on a pattern.

TRIGger:A:BUS:S8B10B:DISParity

Sets or queries the type of disparity that the 8B10B bus will trigger on.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:BUS:S8B10B:DISParity {NEGAtive|POSITIVe|EITher}
TRIGger:A:BUS:S8B10B:DISParity?
```

Related Commands TRIGger:A:BUS:S8B10B:ERROR,

Arguments

NEGative specifies negative disparity.

POSITIVe specifies positive disparity.

EITher specifies either disparity.

Examples

TRIGGER:A:BUS:S8B10B:DISPARITY NEGATIVE sets the disparity type to trigger on to negative.

TRIGGER:A:BUS:S8B10B:DISPARITY? might return :TRIGGER:A:BUS:S8B10B:DISPARITY EITHER, indicating that the oscilloscope will trigger on both negative and positive disparity.

TRIGger:A:BUS:S8B10B:ERROR

Sets or queries whether the oscilloscope will trigger on a character error or a disparity error.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:BUS:S8B10B:ERROR {CHARacter|DISParity}
TRIGger:A:BUS:S8B10B:ERROR?
```

Related Commands TRIGger:A:BUS:S8B10B:DISParity

Arguments

CHARACTER sets the oscilloscope to trigger on a character error.

DISPARITY sets the oscilloscope to trigger on a disparity error.

Examples

TRIGGER:A:BUS:S8B10B:ERROR CHARACTER sets the oscilloscope to trigger on a character error.

TRIGGER:A:BUS:S8B10B:ERROR? might return :TRIGGER:A:BUS:S8B10B:ERROR DISPARITY, indicating that the oscilloscope is set to trigger on a disparity error.

TRIGger:A:BUS:S8B10B:FORMat

Sets or queries the pattern editing format.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:BUS:S8B10B:FORMat {CHAR|SYMbol}
TRIGger:A:BUS:S8B10B:FORMat?
```

Arguments `CHAR` indicates that the pattern editing format is set to character.

`SYMBOL` indicates that the pattern editing format is set to symbol.

Examples `TRIGGER:A:BUS:S8B10B:FORMAT CHAR` sets the pattern editing format to character.

`TRIGGER:A:BUS:S8B10B:FORMAT?` might return `:TRIGGER:A:BUS:S8B10B:FORMAT SYMBOL`, indicating that the pattern editing format is symbol.

TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>

Sets or queries the specified 8B10B character when the Trigger Condition is set to Pattern.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:BUS:S8B10B:PATtern:CHAR<x> <string>
TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>?
```

Related Commands TRIGger:A:BUS:S8B10B:CONDition, TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>, TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>

Arguments `<string>` is the character.

Examples

`TRIGger:A:BUS:S8B10B:PATtern:CHAR1 "K28.5"` sets the first character to K28.5.

`TRIGger:A:BUS:S8B10B:PATtern:CHAR3?` might return `:TRIGger:A:BUS:S8B10B:PATtern:CHAR3 "K28.3"`, indicating that the third character is set to K28.3

TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>

Sets or queries the specified 8B10B character bit pattern (negative disparity) when the Trigger Condition is set to Pattern.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x> <string>
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>?
```

Related Commands TRIGger:A:BUS:S8B10B:CONDition, TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>, TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>

Arguments `<string>` is the character bit pattern.

Examples

`TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:MINUS2 "0011111010"` sets the second character bit pattern to 0011111010.

`:TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:MINUS3?` might return `:TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:MINUS3 0011110011`, indicating that the third character bit pattern is set to 0011110011.

TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>

Sets or queries the specified 8B10B character bit pattern (positive disparity) when the Trigger Condition is set to Pattern.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x> <string>
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>?
```

Related Commands TRIGger:A:BUS:S8B10B:CONDition, TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>, TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>

Arguments <string> is the character bit pattern.

Examples TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:PLUS2 "1100000101" sets the second character bit pattern to 1100000101.

:TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:PLUS3? might return :TRIGGER:A:BUS:S8B10B:PATTERN:SYMBOL:PLUS3 "1100001100", indicating that the third character bit pattern is set to 1100001100.

TRIGger:A:BUS:SOUrce

NOTE. *Not available on some models.*

This command sets or returns the source for a bus trigger.

Group Trigger

Syntax

```
TRIGger:A:BUS:SOUrce
{B1|B2|B3|B4|B5|B6|B7|B8|B9|B10|B11|B12| B13|B14|B15|B16}
TRIGger:A:BUS:SOUrce?
```

Arguments B1–B16 sets the selected source to the bus.

Examples TRIGGER:A:BUS:SOURCE B2 sets the selected source for the bud trigger to Bus 2.

TRIGGER:A:BUS:SOURCE? might return TRIGGER:A:BUS:SOURCE B1, indicating that the selected source for the bus trigger is set to Bus 1.

TRIGger:A:BUS:SPI:CONDition

NOTE. *Not available on some models.*

This command sets or queries the trigger condition for the SPI bus.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:SPI:CONDition {DATA|SS}
TRIGger:A:BUS:SPI:CONDition?
```

Arguments `DATA` sets the trigger condition to Master-In Slave-Out and Master-Out Slave-In.

`SS` sets the trigger condition to Slave selection.

Examples `TRIGGER:A:BUS:SPI:CONDITION SS` sets the trigger condition to Slave selection.

`TRIGGER:A:BUS:SPI:CONDITION?` might return `:TRIGGER:A:BUS:SPI:CONDITION SS`, indicating that the trigger condition is set to Slave selection.

TRIGger:A:BUS:SPI:DATa:FORMat

NOTE. *Not available on some models.*

This command sets or queries the format of the data to be used for an SPI trigger when the trigger condition is Data.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:SPI:DATa:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:SPI:DATa:FORMat?
```

Related Commands TRIGger:A:BUS:SPI:DATa:SIZe, TRIGger:A:BUS:SPI:DATa:VALue

Arguments `BINary` specifies binary as the data format for the SPI bus.

`HEXadecimal` specifies hexadecimal as the data format for the SPI bus.

Examples `TRIGGER:A:BUS:SPI:DATA:FORMAT: BINARY` sets the data format for the SPI bus to binary.

TRIGGER:A:BUS:SPI:DATA:FORMAT? might return :TRIGGER:A:BUS:SPI:DATA:FORMAT BINARY, indicating that the data format for the SPI bus is set to binary.

TRIGger:A:BUS:SPI:DATa:SIZe

NOTE. *Not available on some models.*

This command sets or queries the length of the data string in bytes to be used for an SPI trigger when the trigger condition is Data.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:SPI:DATa:SIZe <NR3>
TRIGger:A:BUS:SPI:DATa:SIZe?
```

Related Commands TRIGger:A:BUS:SPI:DATa:FORMat, TRIGger:A:BUS:SPI:DATa:VALue

Arguments <NR3> specifies the number of contiguous data bytes to trigger on.

Examples TRIGGER:A:BUS:SPI:DATA:SIZE 1 sets the SPI bus trigger data size to one byte.

TRIGGER:A:BUS:SPI:DATA:SIZE? might return :TRIGGER:A:BUS:SPI:DATA:SIZE 2, indicating that the data size for the SPI bus trigger is set to two bytes.

TRIGger:A:BUS:SPI:DATa:VALue

NOTE. *Not available on some models.*

This command sets or queries the data value of the data token for an SPI trigger when the trigger condition is Data.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:SPI:DATa:VALue <string>
TRIGger:A:BUS:SPI:DATa:VALue?
```

Related Commands TRIGger:A:BUS:SPI:DATa:FORMat, TRIGger:A:BUS:SPI:DATa:SIZe

Arguments `<string>` specifies the data value in the specified valid format. The valid characters are 0-9, A-F, and X for hexadecimal format; and 0, 1, and X for a binary number.

Examples `TRIGGER:A:BUS:SPI:DATA:VALUE` "00001111" sets the SPI bus trigger data value to "00001111".

`TRIGGER:A:BUS:SPI:DATA:VALUE?` might return `:TRIGGER:A:BUS:SPI:DATA:VALUE` "01", indicating that the data value for the SPI bus trigger is set to "01".

TRIGger:A:BUS:USB:ADDress:FORMat

NOTE. *Not available on some models.*

This command sets or queries the address format for the USB bus trigger normal token.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:ADDress:FORMat
{BINary|HEXadecimal|SYMBolic}
TRIGger:A:BUS:USB:ADDress:FORMat?
```

Related Commands TRIGger:A:BUS:USB:ADDress:HIVALue, TRIGger:A:BUS:USB:ADDress:VALue

Arguments `TRIGger:A:BUS:USB:ADDress:FORMat BINary` specifies the address format for the normal token as Binary.

`TRIGger:A:BUS:USB:ADDress:FORMat HEXadecimal` specifies the address format for the normal token as Hexadecimal.

`TRIGger:A:BUS:USB:ADDress:FORMat SYMBolic` specifies the address format for the normal token as Symbolic.

Examples

`TRIGGER:A:BUS:USB:ADDRESS:FORMAT SYMBOLIC` sets the address format for the normal token to SYMBolic.

`TRIGGER:A:BUS:USB:ADDRESS:FORMAT?` might return `:TRIGGER:A:BUS:USB:ADDRESS:FORMAT BINARY`, indicating that the address format is set to BINary.

TRIGger:A:BUS:USB:ADDress:HIVALue

NOTE. *Not available on some models.*

This command sets or queries the address value for normal token to be used with In Range and Out of Range qualifiers. The Trigger on Condition is set to Token (Address) Packet.

The VALue and HIVALue set a range that the INrange and OUTrange qualifiers used to decide when to trigger. For example, if the QUALIFER is set to INrange, and the address is within the range set by VALue and HIVALue, then a trigger can be generated.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:ADDress:HIVALue <string>
TRIGger:A:BUS:USB:ADDress:HIVALue?
```

Related Commands

TRIGger:A:BUS:USB:ADDress:VALue

TRIGger:A:BUS:USB:QUAlifier, TRIGger:A:BUS:USB:ADDress:FORMat

Arguments

`<string>` specifies the data value in the specified valid format (binary, hex, or decimal).

Examples

`TRIGGER:A:BUS:USB:ADDRESS:HIVALUE "1111111"` sets the upper limit of the address value for normal token to "1111111".

`TRIGGER:A:BUS:USB:ADDRESS:HIVALUE?` might return `:TRIGGER:A:BUS:USB:ADDRESS:HIVALUE "0001111"`, indicating that the upper limit of the address value for normal token is set to "0001111".

TRIGger:A:BUS:USB:ADDress:VALue

NOTE. *Not available on some models.*

This command sets or queries the address value for normal token.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:ADDress:VALue <string>
TRIGger:A:BUS:USB:ADDress:VALue?
```

Related Commands TRIGger:A:BUS:USB:ADDress:HIVALue

TRIGger:A:BUS:USB:QUALifier, TRIGger:A:BUS:USB:ADDress:FORMat

Arguments `<string>` specifies the data value in the specified valid format (binary, hex, or symbolic).

Examples `TRIGGER:A:BUS:USB:ADDRESS:VALUE` “1110000” sets the address value for the normal token to “1110000”.

`TRIGGER:A:BUS:USB:ADDRESS:VALUE?` might return `:TRIGGER:A:BUS:USB:ADDRESS:VALUE` “0000000”, indicating that the address value for the normal token is set to “0000000”.

TRIGger:A:BUS:USB:CHARacter:CHAR

Sets or queries the USB character when the Trigger Condition is set to Character.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:CHARacter:CHAR <string>
TRIGger:A:BUS:USB:CHARacter:CHAR?
```

Arguments <string> is the character bit pattern.

Examples `TRIGGER:A:BUS:USB:CHARACTER:CHAR "0011111010"` will trigger on the 0011111010 character.

`TRIGGER:A:BUS:USB:CHARACTER:CHAR?` may return "1100000101" indicating the instrument will trigger on the 1100000101 character.

TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus

Sets or queries the character bit pattern (negative disparity) when the Trigger Condition is set to Character.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus <string>
TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus?
```

Arguments <string> is the character bit pattern.

Examples `TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:MINUS "0011111010"` sets the character bit pattern to 0011111010.

`TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:MINUS?` might return `:TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:MINUS "0011110011"`, indicating that the character bit pattern is set to 0011110011.

TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS

Sets or queries the Character bit pattern (positive disparity view) when the Trigger Condition is set to Character.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS <string>
TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS?
```

Arguments <string> is the character bit pattern.

Examples `TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:PLUS` "1100000101" sets the character bit pattern to 1100000101.

`TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:PLUS?` might return `:TRIGGER:A:BUS:USB:CHARACTER:SYMBOL:PLUS "1100001100"`, indicating that the character bit pattern is 1100001100.

TRIGger:A:BUS:USB:CONDition

NOTE. *Not available on some models.*

This command sets or returns the condition for a USB trigger, where x is the Trigger on condition..

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:CONDition
{DATAPacket|EOP|ERROR|HANDSHAKEPacket|RESET|RESUME|
SPECIALPacket|SUSPEND|SYNC|TOKENPacket}
TRIGger:A:BUS:USB:CONDition?
```

Arguments

`DATAPacket` sets the Trigger on condition to Data Packet.

`EOP` sets the Trigger on condition to End of Packet.

`ERROR` sets the Trigger on condition to Error.

`HANDSHAKEPacket` sets the Trigger on condition to Handshake Packet.

`RESET` sets the Trigger on condition to Reset.

`RESUME` sets the Trigger on condition to Resume.

`SPECIALPacket` sets the Trigger on condition to Special Packet.

`SUSPEND` sets the Trigger on condition to Suspend.

`SYNC` sets the Trigger on condition to Sync.

`TOKENPacket` sets the Trigger on condition to Token (Address) Packet.

Examples `TRIGGER:A:BUS:USB:CONDITION RESET` sets the USB trigger on condition to Reset.

`TRIGGER:A:BUS:USB:CONDITION?` might return `:TRIGGER:A:BUS:USB:CONDITION SYNC`, indicating that the USB trigger on condition is set to Sync.

TRIGger:A:BUS:USB:DATa:FORMat

This command sets or queries the data format for the USB bus trigger.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:DATa:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:USB:DATa:FORMat?
```

Related Commands TRIGger:A:BUS:USB:DATa:HIVALue

Arguments `BINary` specifies the data format for the USB bus trigger as Binary.

`HEXadecimal` specifies the data format for the USB bus trigger as Hexadecimal.

Examples `TRIGGER:A:BUS:USB:DATA:FORMAT BINARY` sets the data format to binary.

`TRIGGER:A:BUS:USB:DATA:FORMAT ?` might return `:TRIGGER:A:BUS:USB:DATA:FORMAT HEX`, indicating that the data format is set to hexadecimal.

TRIGger:A:BUS:USB:DATa:HIVALue

NOTE. *Not available on some models.*

This command sets or queries data value for data token to be used with In Range and Out of Range qualifiers. The Trigger on Condition is set to Data Packet.

The VALue and HIVALue set a range that the INrange and OUTrange qualifiers used to decide when to trigger. For example, if the QUALIFER is set to INrange, and the address is within the range set by VALue and HIVALue, then a trigger can be generated.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:DATa:HIVALue <string>
TRIGger:A:BUS:USB:DATa:HIVALue?
```

Related Commands TRIGger:A:BUS:USB:DATa:VALue

TRIGger:A:BUS:USB:QUALifier

Arguments <string> specifies the data value in the specified valid format (binary, hex, or decimal).

Examples TRIGGER:A:BUS:USB:DATA:HIVALUE "00001000" sets the upper limit of the data value for data token to 00001000 (08 hex).

TRIGGER:A:BUS:USB:DATA:HIVALUE? might return :TRIGGER:A:BUS:USB:DATA:HIVALUE "01111111", indicating that the upper limit of the data value for data token is set to 01111111 (7F hex).

TRIGger:A:BUS:USB:DATa:OFFSet

NOTE. *Not available on some models.*

This command sets or queries data offset for the USB bus trigger.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:DATa:OFFSet {<NR3>|DONTCare}
TRIGger:A:BUS:USB:DATa:OFFSet?
```

Arguments <NR3> specifies the data offset in bytes.

DONTCare specifies the data offset as Don't Care.

Examples TRIGGER:A:BUS:USB:DATA:OFFSET2 sets two bytes data offset for the USB bus trigger.

TRIGGER:A:BUS:USB:DATA:OFFSET? might return :TRIGGER:A:BUS:USB:DATA:OFFSET 1, indicating that the data offset for the USB bus trigger is set to 1 byte.

TRIGger:A:BUS:USB:DATa:SIZe

NOTE. *Not available on some models.*

This command sets or queries the number of contiguous data bytes to USB bus trigger.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:DATa:SIZe <NR3>
TRIGger:A:BUS:USB:DATa:SIZe?
```

Arguments <NR3> specifies the data size in bytes.

Examples TRIGGER:A:BUS:USB:DATA:SIZE 3 sets three bytes data size for the USB bus trigger.

TRIGGER:A:BUS:USB:DATA:SIZE? might return :TRIGGER:A:BUS:USB:DATA:SIZE 1, indicating that the data size for the USB bus trigger is set to 1 byte.

TRIGger:A:BUS:USB:DATa:TYPe

NOTE. *Not available on some models.*

This command sets or queries the data packet type when USB bus trigger condition is Data Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:DATa:TYPe {ANY|DATA0|DATA1|DATA2|MDATA}
TRIGger:A:BUS:USB:DATa:TYPe?
```

Arguments ANY specifies the data packet type to ANY.

DATA0 specifies the data packet type to DATA0.

DATA1 specifies the data packet type to DATA1.

DATA2 specifies the data packet type to DATA2.

MDATA specifies the data packet type to MDATA.

Examples `TRIGGER:A:BUS:USB:DATA:TYPEMDATA` sets the data packet type to MDATA.

`TRIGGER:A:BUS:USB:DATA:TYPE?` might return `:TRIGGER:A:BUS:USB:DATA:TYPE ANY`, indicating that the data packet type is set to ANY when the USB bus trigger on condition is Data Packet.

TRIGger:A:BUS:USB:DATa:VALue

NOTE. *Not available on some models.*

This command sets or queries the USB bus trigger data value for data token.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:DATa:VALue <string>
TRIGger:A:BUS:USB:DATa:VALue?
```

Related Commands TRIGger:A:BUS:USB:DATa:HIVALue

TRIGger:A:BUS:USB:QUAlifier

Arguments `<string>` specifies the data value. The valid characters are 0, 1, or X representing a binary number.

Examples `TRIGGER:A:BUS:USB:DATA:VALUE "00001111"` sets the USB bus trigger data value for data token to "00001111".

`TRIGGER:A:BUS:USB:DATA:VALUE?` might return `:TRIGGER:A:BUS:USB:DATA:VALUE "XXXX"`, indicating that the USB bus trigger data value is "XXXX".

TRIGger:A:BUS:USB:DISParity

Sets or queries the type of disparity that the bus will trigger on.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:DISParity {NEGAtive|POSITIVe|EITher}
TRIGger:A:BUS:USB:DISParity?
```

Arguments

`NEGative` specifies negative disparity.

`POSITIVe` specifies positive disparity.

`EITher` specifies either disparity.

Examples

`TRIGGER:A:BUS:USB:DISPARITY NEGATIVE` set the instrument to trigger on negative disparity.

`TRIGGER:A:BUS:USB:DISPARITY?` may return `POSITIVE` indicating the instrument will trigger on positive disparity.

TRIGger:A:BUS:USB:ENDPoint:FORMat

NOTE. *Not available on some models.*

This command sets or queries the endpoint format for data token to be used with in range and out of range qualifiers.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:ENDPoint:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:USB:ENDPoint:FORMat?
```

Related Commands

TRIGger:A:BUS:USB:ENDPoint:HIVALue, TRIGger:A:BUS:USB:ENDPoint:VALue

Arguments

`BINary` specifies the endpoint format for data token as Binary.

`HEXadecimal` specifies the endpoint format for data token as Hexadecimal

Examples

`TRIGGER:A:BUS:USB:ENDPOINT:FORMAT BINARY` sets the endpoint format for the data token to binary.

`TRIGGER:A:BUS:USB:ENDPOINT:FORMAT?` might return `:TRIGGER:A:BUS:USB:ENDPOINT:FORMAT SYMBOLIC`, indicating that the endpoint format for the data token is set to symbolic.

TRIGger:A:BUS:USB:ENDPoint:HIVALue

NOTE. *Not available on some models.*

This command sets or queries the endpoint value for data token to be used with in range and out of range qualifiers. The VALue and HIVALue set a range that the INrange and OUTrange qualifiers used to decide when to trigger. For example, if the QUALIFIER is set to INrange, and the address is within the range set by VALue and HIVALue, then a trigger can be generated.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:ENDPoint:HIVALue <string>
TRIGger:A:BUS:USB:ENDPoint:HIVALue?
```

Related Commands TRIGger:A:BUS:USB:ENDPoint:VALue

TRIGger:A:BUS:USB:QUALifier, TRIGger:A:BUS:USB:ENDPoint:FORMat

Arguments `<string>` specifies the endpoint value in the specified valid format (binary, hex, or symbolic).

Examples `TRIGGER:A:BUS:USB:ENDPOINT:HIVALUE "0010"` sets the upper limit of the endpoint value for data token to "0010".

`TRIGGER:A:BUS:USB:ENDPOINT:HIVALUE?` might return `:TRIGGER:A:BUS:USB:ENDPOINT:HIVALUE "XXXXXX01"`, indicating that the USB bus trigger endpoint value for data token is "XXXXXX01".

TRIGger:A:BUS:USB:ENDPoint:VALue

NOTE. *Not available on some models.*

This command sets or queries the endpoint value for normal token.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:ENDPoint:VALue <string>
TRIGger:A:BUS:USB:ENDPoint:VALue?
```

Related Commands TRIGger:A:BUS:USB:QUAlifier, TRIGger:A:BUS:USB:ENDPoint:HIVALue, TRIGger:A:BUS:USB:ENDPoint:FORMat

Arguments <string> specifies the data value in the specified valid format (binary, hex, or symbolic).

Examples TRIGGER:A:BUS:USB:ENDPOINT:VALUE "11111111" sets the endpoint value for normal token to "11111111".

TRIGGER:A:BUS:USB:ENDPOINT:VALUE? might return :TRIGGER:A:BUS:USB:ENDPOINT:VALUE "1111", indicating that the USB bus trigger endpoint value for data token is "1111".

TRIGger:A:BUS:USB:ERROR

Sets or queries whether the oscilloscope will trigger on a character error or a disparity error.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:ERROR {CHARacter|DISParity}
TRIGger:A:BUS:USB:ERROR?
```

Arguments CHARACTER sets the oscilloscope to trigger on a character error.

DISPARITY sets the oscilloscope to trigger on a disparity error.

Examples TRIGGER:A:BUS:USB:ERROR CHAR sets the oscilloscope to trigger on a character error.

TRIGGER:A:BUS:USB:ERROR? may return DISP indicating that the instrument is set to trigger on a disparity error.

TRIGger:A:BUS:USB:ERRTYPE

NOTE. *Not available on some models.*

This command sets or queries the error type when USB bus trigger on condition is set to Error.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:ERRTYPE {BITSTUFFing|CRC5|CRC16|PID}
TRIGger:A:BUS:USB:ERRTYPE?
```

Arguments

`BITSTUFFing` specifies the error type Bit Stuffing .

`CRC5` specifies the error type as Token CRC5 (Cyclic Redundancy Check 5).

`CRC16` specifies the error type as Data CRC16 (Cyclic Redundancy Check 16).

`PID` specifies the error type as PID Check Bits.

Examples

`TRIGGER:A:BUS:USB:ERRTYPE CRC16` sets the USB bus trigger on to Data CRC16 error type.

`TRIGGER:A:BUS:USB:ERRTYPE?` might return `:TRIGGER:A:BUS:USB:ERRTYPE PID`, indicating that the USB bus trigger on is set to PID Check Bits error type.

TRIGger:A:BUS:USB:FORMat

Sets or queries the pattern editing format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:FORMat {CHAR|SYMbol}
TRIGger:A:BUS:USB:FORMat?
```

Arguments

`CHAR` indicates that the pattern editing format is set to character.

`SYMBOL` indicates that the pattern editing format is set to symbol.

Examples

`TRIGGER:A:BUS:USB:FORMAT SYMBOL` indicates the pattern editing format is set to SYMBOL.

`TRIGGER:A:BUS:USB:FORMAT?` may return `CHAR` indicating that the pattern editing format is set to character.

TRIGger:A:BUS:USB:HANDShaketype

NOTE. *Not available on some models.*

This command sets or queries the handshake type when USB bus trigger on condition is set to Handshake Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:HANDShaketype {ACK|ANY|NAK|NYET|STALL}
TRIGger:A:BUS:USB:HANDShaketype?
```

Arguments

`ACK` specifies the handshake type as Acknowledge (XX10).

`ANY` specifies the handshake type as Any (0010).

`NAK` specifies the handshake type as Negative Acknowledge (1010).

`NYET` specifies the handshake type as No response Yet (0110).

`STALL` specifies the handshake type as Stall (1110).

Examples

`TRIGGER:A:BUS:USB:HANDSHAKETYPE NAK` sets the handshake type to NAK.

`TRIGGER:A:BUS:USB:HANDSHAKETYPE?` might return `:TRIGGER:A:BUS:USB:HANDSHAKETYPE ANY`, indicating that the handshake type is set to ANY.

TRIGger:A:BUS:USB:PATtern:CHAR<x>

Sets or queries the specified character when the Trigger Condition is set to Pattern.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:PATtern:CHAR<x> <string>
TRIGger:A:BUS:USB:PATtern:CHAR<x>?
```

Arguments `<string>` is the specified character.

Examples `TRIGger:A:BUS:USB:PATtern:CHAR2“0011111010”` will trigger on the 0011111010 character when it is character 2..

`TRIGger:A:BUS:USB:PATtern:CHAR1?` may return “1100000101” indicating the instrument will trigger on 1100000101 when it is character 1.

TRIGger:A:BUS:USB:PATtern:NUMSymbols

Sets or queries the number of symbols to trigger on when USB trigger type is in pattern mode.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:PATtern:NUMSymbols <NR1>
TRIGger:A:BUS:USB:PATtern:NUMSymbols?
```

Arguments <NR1> is the number 2, 3, or 4..

Examples `TRIGGER:A:BUS:USB:PATTERN:NUMSYMBOLS 2` sets the number of symbols to 2.

`TRIGGER:A:BUS:USB:PATTERN:NUMSYMBOLS?` might return `:TRIGGER:A:BUS:USB:PATTERN:NUMSYMBOLS 2`.

TRIGger:A:BUS:USB:PATtern:ORDERedset

This command sets or queries the state of triggering on an ordered set.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:PATtern:ORDERedset {OFF|ON|0|1}
TRIGger:A:BUS:USB:PATtern:ORDERedset?
```

Arguments `OFF` disables triggering on an ordered set.

`ON` enables triggering on an ordered set.

`0` disables triggering on an ordered set.

`1` enables triggering on an ordered set.

Examples `TRIGGER:A:BUS:USB:PATTERN:ORDEREDSET ON` enables triggering on an ordered set.

`TRIGGER:A:BUS:USB:PATTERN:ORDEREDSET?` may return `OFF` indicating that triggering on an ordered set is disabled.

TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>

Sets or queries the specified character bit pattern (negative disparity) when the Trigger Condition is set to Pattern.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x> <string>
TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>?
```

Arguments `<string>` is the specified character bit pattern.

Examples `TRIGger:A:BUS:USB:PATtern:SYMbol:MINus3 "0011111010"` sets the third character bit pattern to 0011111010.

`TRIGger:A:BUS:USB:PATtern:SYMbol:MINus2?` might return `:TRIGGER:A:BUS:USB:PATTERN:SYMBOL:MINUS2 "0011110011"`, indicating that the second character bit pattern is set to 0011110011.

TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>

Sets or queries the specified character bit pattern (positive disparity) when the Trigger Condition is set to Pattern.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x> <string>
TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>?
```

Arguments `<string>` is the specified character bit pattern.

Examples

`TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS3“0011111010”` sets the third character bit pattern to 0011111010.

`TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS2?` might return `:TRIGGER:A:BUS:USB:PATTERN:SYMBOL:PLUS2 “0011110011”`, indicating that the second character bit pattern is set to 0011110011.

TRIGger:A:BUS:USB:QUAlifier

NOTE. *Not available on some models.*

This command sets or queries the USB bus trigger qualifier for address, endpoint, and data.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:QUAlifier
{EQUal|INrange|LESSEQual|MOREEQual|OUTrange|UNEQual|
LESSThan|MOREThan}
TRIGger:A:BUS:USB:QUAlifier?
```

Related Commands

TRIGger:A:BUS:USB:ADDress:HIVALue

TRIGger:A:BUS:USB:ADDress:VALue

TRIGger:A:BUS:USB:DATa:HIVALue

TRIGger:A:BUS:USB:DATa:VALue

TRIGger:A:BUS:USB:ENDPoint:HIVALue

TRIGger:A:BUS:USB:ENDPoint:VALue

Arguments

`EQUal` specifies = trigger qualifier.

`INrange` specifies Inside Range trigger qualifier.

`LESSEQual` specifies <= trigger qualifier.

`MOREEQual` specifies >= trigger qualifier.

`OUTrange` specifies = trigger qualifier.

`UNEQual` specifies != trigger qualifier.

`LESSThan` specifies < trigger qualifier.

`MOREThan` specifies > trigger qualifier.

Examples `TRIGGER:A:BUS:USB:QUALIFIER EQUAL` sets the trigger qualifier to EQUal.

`TRIGGER:A:BUS:USB:QUALIFIER?` might return `:TRIGGER:A:BUS:USB:QUALIFIER INRANGE`, indicating that the trigger qualifier is INrange.

TRIGger:A:BUS:USB:SOF:FORMat

NOTE. *Not available on some models.*

This command sets or queries the format for the Start of Frame frame number.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:SOF:FORMat {BINary|HEXadecimal|SYMBolic}
TRIGger:A:BUS:USB:SOF:FORMat?
```

Related Commands TRIGger:A:BUS:USB:SOF:FRAMENUMber

Arguments `BINary` specifies the Start of Frame frame number format as Binary.

`HEXadecimal` specifies the Start of Frame frame number format as Hexadecimal.

`SYMBolic` specifies the Start of Frame frame number format as Symbolic.

Examples `TRIGGER:A:BUS:USB:SOF:FORMAT SYMBOLIC` sets the SOF frame number format to symbolic.

`TRIGGER:A:BUS:USB:SOF:FORMAT?` might return `:TRIGGER:A:BUS:USB:SOF:FORMAT BINARY`, indicating that the format of the SOF frame number is set to binary.

TRIGger:A:BUS:USB:SOF:FRAMENUMber

NOTE. *Not available on some models.*

This command sets frame number for the USB bus trigger when the condition is Start of Frame (SOF).

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SOF:FRAMENUMber <string>
TRIGger:A:BUS:USB:SOF:FRAMENUMber?
```

Related Commands TRIGger:A:BUS:USB:SOF:FORMat

Arguments <string> specifies the frame number for the SOF in the specified valid format.

Examples TRIGGER:A:BUS:USB:SOF:FRAMENUMBER "01" sets the frame number to 01.

TRIGGER:A:BUS:USB:SOF:FRAMENUMBER? might return :TRIGGER:A:BUS:USB:SOF:FRAMENUMBER "XXXXXXXXX01", indicating that the frame number is set to 01, in the format of binary.

TRIGger:A:BUS:USB:SPECIALType

NOTE. *Not available on some models.*

This command sets or queries the PID value when USB bus Trigger on condition is set to Special Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SPECIALType
{ANY|ERR|PING|PRE|RESERVED|SPLIT}
TRIGger:A:BUS:USB:SPECIALType?
```

Arguments

ANY specifies Any (XX00) PID value.

ERR specifies ERR (1100) PID value.

PING specifies PING (0100) PID value.

PRE specifies PRE (1100) PID value.

RESERVED specifies Reserved (0000) PID value.

SPLIT specifies Split (1000) PID value.

Examples TRIGGER:A:BUS:USB:SPECIALTYPE Reserved sets the PID value to Reserved (0000).

TRIGGER:A:BUS:USB:SPECIALTYPE? might return :TRIGGER:A:BUS:USB:SPECIALTYPE ANY, indicating that the PID value is set to ANY (XX00).

TRIGger:A:BUS:USB:SPLIT:ET:VALue

NOTE. *Not available on some models.*

This command sets or queries the Endpoint Type value for SPLIT token. The USB Speed is set to High (480 Mbps) and the Trigger condition is set to Special Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SPLIT:ET:VALue
{DONTcare|CONTRol|ISOchronous|BULK|INTERRUPT}
TRIGger:A:BUS:USB:SPLIT:ET:VALue?
```

Arguments

DONTCare specifies the End Point Type value as don't care (XX).

CONTRol specifies the End Point Type value as Control (00).

ISOchronous specifies the End Point Type value as Isochronous (01).

BULK specifies the End Point Type value as Bulk (10).

INTERRUPT specifies the End Point Type value as Interrupt (11).

Examples

TRIGGER:A:BUS:USB:SPLIT:ET:VALUE control sets the End Point Type value to Control.

TRIGGER:A:BUS:USB:SPLIT:ET:VALUE? might return :TRIGGER:A:BUS:USB:SPLIT:ET:VALUE DONTCARE, indicating that the End Point Type value is set to DONTCare.

TRIGger:A:BUS:USB:SPLIT:HUB:FORMat

This command sets or queries the format for the hub address of the USB bus trigger on split token field.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SPLIT:HUB:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:USB:SPLIT:HUB:FORMat?
```

Related Commands TRIGger:A:BUS:USB:SPLIT:HUB:VALue

Arguments

`BINary` specifies hub address format as Binary.

`HEXadecimal` specifies hub address format as Hexadecimal.

Examples

`TRIGGER:A:BUS:USB:SPLIT:HUB:FORMAT BINARY` sets the format for the hub address to binary.

`TRIGGER:A:BUS:USB:SPLIT:HUB:FORMAT?` might return `:TRIGGER:A:BUS:USB:SPLIT:HUB:FORMAT HEX`, indicating that the format for the hub address is set to hexadecimal.

TRIGger:A:BUS:USB:SPLIT:HUB:VALue

NOTE. *Not available on some models.*

This command sets or queries the hub address of the USB bus trigger on split token field. The USB Speed is set to High (480 Mbps) and the Trigger condition is set to Special Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SPLIT:HUB:VALue <string>
TRIGger:A:BUS:USB:SPLIT:HUB:VALue?
```

Related Commands TRIGger:A:BUS:USB:SPLIT:HUB:FORMat

Arguments

`<string>` specifies the hub address in the specified valid format.

Examples

`TRIGGER:A:BUS:USB:SPLIT:HUB:VALUE "0001"` sets the hub address to "0001".

`TRIGGER:A:BUS:USB:SPLIT:HUB:VALUE?` might return `:TRIGGER:A:BUS:USB:SPLIT:HUB:VALUE "XX"`, indicating that the hub address is set to "XX" and is in hexadecimal format.

TRIGger:A:BUS:USB:SPLIT:PORT:FORMat

NOTE. *Not available on some models.*

This command sets the port address format for the USB bus trigger on split token field.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SPLIT:PORT:FORMat {BINary|HEXadecimal}
TRIGger:A:BUS:USB:SPLIT:PORT:FORMat?
```

Related Commands TRIGger:A:BUS:USB:SPLIT:HUB:VALue

Arguments `BINary` sets the port address format to Binary.

`HEXadecimal` sets the port address format to Hexadecimal.

Examples `TRIGGER:A:BUS:USB:SPLIT:PORT:FORMAT HEXADECIMAL` sets the port address format to hexadecimal.

`TRIGGER:A:BUS:USB:SPLIT:PORT:FORMAT?` might return `:TRIGGER:A:BUS:USB:SPLIT:PORT:FORMAT BINARY`, indicating that the port address format is set to binary.

TRIGger:A:BUS:USB:SPLIT:PORT:VALue

NOTE. *Not available on some models.*

This command sets or queries the port address for the USB bus trigger on split token field. The USB Speed is set to High (480 Mbps) and the Trigger condition is set to Special Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SPLIT:PORT:VALue <string>
TRIGger:A:BUS:USB:SPLIT:PORT:VALue?
```

Related Commands TRIGger:A:BUS:USB:SPLIT:PORT:FORMat

Arguments <string> specifies the port address in the specified valid format.

Examples TRIGGER:A:BUS:USB:SPLIT:PORT:VALUE "1111" sets the port address for SPLIT token to "1111".

TRIGGER:A:BUS:USB:SPLIT:PORT:VALUE? might return :TRIGGER:A:BUS:USB:SPLIT:PORT:VALUE "XX", indicating that the port address for SPLIT token is set to "XX".

TRIGger:A:BUS:USB:SPLIT:SC:VALue

NOTE. *Not available on some models.*

This command sets or queries the Start/Complete value for the USB bus trigger on split token field. The USB Speed is set to High (480 Mbps) and the Trigger condition is set to Special Packet.

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:BUS:USB:SPLIT:SC:VALue {CSPLIT|SSPLIT|DONTcare}
TRIGger:A:BUS:USB:SPLIT:SC:VALue?
```

Arguments DONTcare specifies the Start/Complete value as don't care (X).

SSPLIT specifies the Start/Complete value as Start (SSPLIT) (0).

CSPLIT specifies the Start/Complete value as Complete (CSPLIT)(1).

Examples TRIGGER:A:BUS:USB:SPLIT:SC:VALUE csplit sets the Start/Complete value to CSPLIT.

TRIGGER:A:BUS:USB:SPLIT:SC:VALUE? might return :TRIGGER:A:BUS:USB:SPLIT:SC:VALUE DONTCARE, indicating that the Start/Complete value is DONTCare.

TRIGger:A:BUS:USB:SPLIT:SE:VALue

NOTE. *Not available on some models.*

This command sets or queries the Start/End value for the USB bus trigger on split token field. The USB Speed is set to High (480 Mbps) and the Trigger condition is set to Special Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:SPLIT:SE:VALue
{DONTCare|FULLSPeed|LOWSPeed|ISOMID|ISOEND|ISOSTART|ISOALL}
TRIGger:A:BUS:USB:SPLIT:SE:VALue?
```

Arguments

`DONTCARE` specifies the Start/End bit value as don't care (X).

`FULLSPeed` specifies the Start/End bit value as Control/Bulk/Interrupt Full Speed device (0X).

`LOWSPeed` specifies the Start/End bit value as Control/Bulk/Interrupt Low Speed device (1X).

`ISOMID` specifies the Start/End bit value as Isochronous Data is Middle (00).

`ISOEND` specifies the Start/End bit value as Isochronous Data is End (01).

`ISOSTART` specifies the Start/End bit value as Isochronous Data is Start (10).

`ISOALL` specifies the Start/End bit value as Isochronous Data is All (11).

Examples

`TRIGGER:A:BUS:USB:SPLIT:SE:VALUE ISOALL` sets the start/end bit value to Isochronous Data is All (11).

`TRIGGER:A:BUS:USB:SPLIT:SE:VALUE?` might return `:TRIGGER:A:BUS:USB:SPLIT:SE:VALUE DONTCARE`, indicating that the start/end bit value is set to don't care (X).

TRIGger:A:BUS:USB:TOKENType

NOTE. *Not available on some models.*

This command sets or queries the Token Type when USB Trigger condition is set to Token (Address) Packet.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:BUS:USB:TOKENType {ANY|IN|OUT|SOF|SETUP}
TRIGger:A:BUS:USB:TOKENType?
```

Arguments

`ANY` specifies ANY (XX01) token type.

`IN` specifies IN (1001) token type.

`OUT` specifies OUT (0001) token type.

`SOF` specifies SOF (0101) token type.

`SETUP` specifies SETUP (1101) token type.

Examples

`TRIGGER:A:BUS:USB:TOKENTYPE in` sets the token type to IN (1001).

`TRIGGER:A:BUS:USB:TOKENTYPE?` might return `:TRIGGER:A:BUS:USB:TOKENTYPE ANY`, indicating that the token type is set to ANY.

TRIGger:A:CAN:CONDition

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or returns the CAN condition.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:CONDition
{SOF|FRAMEtype|IDENTifier|DATA|IDANDDATA|EOF|ACKMISS}
TRIGger:A: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.

`ERROR` enables triggering on an error within a frame.

Examples `TRIGGER:A:CAN:CONDITION DATA` enables triggering on matching CAN data.

`TRIGGER:A:CAN:CONDITION?` might return `:TRIGGER:A:CAN:CONDITION EOF`, indicating that the triggering is set on the end of frame.

TRIGger:A:CAN:DATa:DIRection

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the CAN trigger condition to be valid on a READ, WRITE, or either.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:DATa:DIRection {READ|WRITE|NOCARE}
TRIGger:A:CAN:DATa:DIRection?
```

Arguments `READ` sets the CAN data direction to Read.

`WRITE` sets the CAN data direction to Write.

`NOCARE` sets the CAN data direction to either.

Examples `TRIGGER:A:CAN:DATA:DIRECTION WRITE` sets the CAN data direction to Write.

`TRIGGER:A:CAN:DATA:DIRECTION?` might return `:TRIGGER:A:CAN:DATA:DIRECTION READ`, indicating that the CAN data direction is set to Read.

TRIGger:A:CAN:DATa:LEVel

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the CAN Trigger threshold for the CAN data source.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:DATa:LEVel <NR3>
TRIGger:A:CAN:DATa:LEVel?
```

Arguments

<NR3> specifies the CAN trigger data level. The level can be ±12.0 divisions.

Examples

TRIGGER:A:CAN:DATA:LEVEL 0.0 sets the CAN trigger data level to 0.0 Volts.

TRIGGER:A:CAN:DATA:LEVEL? might return :TRIGGER:A:CAN:DATA:LEVEL 5.0000, indicating that the CAN trigger data level is set to 5.0 volts.

TRIGger:A:CAN:DATa:SOUrce

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the CAN data source.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:DATa:SOUrce CH<x>
TRIGger:A:CAN:DATa:SOUrce?
```

Arguments

CH<x> specifies the source for the CAN trigger. x can be 1, 2, 3, or 4.

Examples

TRIGGER:A:CAN:DATA:SOURCE CH1 sets the CAN trigger data source to channel CH1.

TRIGGER:A:CAN:DATA:SOURCE? might return :TRIGGER:A:CAN:DATA:SOURCE CH4, indicating that the CAN trigger data source is set to channel CH4.

TRIGger:A:CAN:DATa:VALue

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or quires the binary data string used for CAN Trigger if the trigger condition is ID or IDANDDATA.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:DATa:VALue <QString>
TRIGger:A:CAN:DATa:VALue?
```

Arguments

<QString> is up to 32 bits specifying the CAN data value.

Examples

TRIGGER:A:CAN:DATA:VALUE "1011" sets the CAN data value to 1011.

TRIGGER:A:CAN:DATA:VALUE? might return :TRIGGER:A:CAN:DATA:SOURCE 10111, indicating that the CAN data value is set to 10111.

TRIGger:A:CAN:FORMat

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the CAN data format.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:FORMat {BINary|HEX}
TRIGger:A:CAN:FORMat?
```

Arguments

BINary specifies binary as the CAN data format.

HEX specifies hexadecimal as the CAN data format.

Examples

TRIGGER:A:CAN:FORMAT BINARY sets the CAN trigger data format to binary.

TRIGGER:A:CAN:FORMAT? might return :TRIGGER:A:CAN:DATA:FORMAT HEX, indicating that the CAN data format is set to hexadecimal.

TRIGger:A:CAN:FRAMEtype

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the CAN trigger frame type.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:FRAMEtype {DATA|REMote|ERROR|OVERLOAD}
TRIGger:A: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:CAN:FRAMETYPE DATA` sets the CAN trigger frame type to data.

`TRIGGER:A:CAN:FRAMETYPE?` might return `:TRIGGER:A:CAN:DATA:FRAMETYPE REMOTE`, indicating that the CAN trigger frame type is set to remote.

TRIGger:A:CAN:IDENTifier:MODe

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the CAN trigger identifier mode.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:IDENTifier:MODe {STANdard|EXTENded}
TRIGger:A:CAN:IDENTifier:MODe?
```

Arguments

`STANdard` sets the CAN trigger identifier mode to standard mode.

`EXTENded` sets the CAN trigger identifier mode to extended mode.

Examples

`TRIGGER:A:CAN:IDENTIFIER:MODE STANDARD` sets the CAN trigger identifier mode to standard.

`TRIGGER:A:CAN:IDENTIFIER:MODE?` might return `:TRIGGER:A:CAN:IDENTIFIER:MODE EXECUTE`, indicating that the CAN trigger identifier mode is set to extended.

TRIGger:A:CAN:IDENTifier:VALue

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the binary address string used for the CAN trigger if the trigger condition is ID or IDANDDATA.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:IDENTifier:VALue <QString>
TRIGger:A:CAN:IDENTifier:VALue?
```

Arguments <QString> is up to 29 bits specifying the CAN identifier value.

Examples TRIGGER:A:CAN:IDENTIFIER:VALUE "1011" sets the CAN trigger identifier value to 1011.

TRIGGER:A:CAN:IDENTIFIER:VALUE? might return :TRIGGER:A:CAN:IDENTIFIER:VALUE "10111", indicating that the CAN trigger identifier value is set to 10111.

TRIGger:A:CAN:PROBE

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the probing method used to probe the CAN signal.

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:PROBE {CANL|CANH|DIFFerential|TX|RX}
TRIGger:A:CAN:PROBE?
```

Arguments CANL specifies that the probing method is CANL.

CANH specifies that the probing method is CANH.

DIFFerential specifies that the probing method used is differential.

TX specifies that the probing method is TX.

RX specifies that the probing method is RX.

Examples

TRIGGER:A:CAN:PROBE CANH sets the method used to probe the CAN signal to CANH.

TRIGGER:A:CAN:PROBE? might return TRIGGER:A:CAN:PROBE DIFFERENTIAL, indicating that the probing method is differential.

TRIGger:A:CAN:SPEed

NOTE. *This trigger is available only when the Option LSA is ordered.*

CAN option only: This command sets or queries the bit rate of the CAN system.

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:CAN:SPEed <NR3>
TRIGger:A:CAN:SPEed?
```

Arguments

<NR3> specifies the bit rate of the CAN system. Possible values are 1M, 800K, 500K, 250K, 125K, 100K, 83.3K, 62,5K, 50K, 33K, 20K, and 10K.

Examples

TRIGGER:A:CAN:SPEED 33K sets the CAN systems with a bit rate of 33K.

TRIGGER:A:CAN:SPEED? might return TRIGGER:A:CAN:SPEED 50K, indicating that the bit rate of the CAN system is set to 50K.

TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:PULSEForm

This entry covers five separate commands, one each for AMI and AMI subtypes. The commands set or query the AMI pulse form to one of three possibilities.

Conditions

This command is only valid if the instrument has Option MTM installed.

Group

Trigger

Syntax

```
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:PULSEForm
{PLUSOne|MINUSOne| EYEdiagram}
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:PULSEForm?
```

Arguments PLUSOne corresponds to the Isolated +1 on the front panel menu.

MINUSOne corresponds to the Isolated -1 on the front panel menu.

EYEDiagram positions the trigger point off screen to display the eye.

Examples TRIGGER:A:COMMUNICATION:AMI:PULSEFORM PLUSONE sets the pulse form to PLUSONE.

TRIGGER:A:COMMUNICATION:AMI:PULSEFORM? might return :TRIGGER:A:COMMUNICATION:AMI:PULSEFORM PLUSONE, indicating that the AMI PULSFORM is set to PLUSONE.

TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:HIGH

This entry covers five separate commands, one each for AMI and AMI subtypes. This command sets or queries the threshold high level.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Trigger

Syntax
```
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:HIGH <NR3>
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:HIGH?
```

Arguments <NR3> is the high threshold parameter expressed in volts.

Examples TRIGGER:A:COMMUNICATION:AMI:THRESHOLD:HIGH 2.8E-2 sets the threshold high level to 2.8E-2.

TRIGGER:A:COMMUNICATION:AMI:THRESHOLD:HIGH? might return :TRIGGER:A:COMMUNICATION:AMI:THRESHOLD:HIGH 2.8E-2, indicating the threshold high level in volts.

TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:LOW

This entry covers five separate commands, one each for AMI and AMI subtypes. This command sets or queries the threshold low level.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Trigger

Syntax
```
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:LOW <NR3>
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:LOW?
```

Arguments <NR3> specifies the high threshold parameter, expressed in volts.

Examples TRIGGER:A:COMMUNICATION:AMI:THRESHOLD:LOW 2.8E-2 sets the threshold low level to 2.8E-2.

TRIGGER:A:COMMUNICATION:AMI:THRESHOLD:LOW? might return :TRIGGER:A:COMMUNICATION:AMI:THRESHOLD:LOW 2.8E-2, indicating the threshold low level in volts.

TRIGger:A:COMMunication:BITRate

NOTE. *Not available on some models.*

This command sets or queries the bit rate.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Trigger

Syntax
```
TRIGger:A:COMMunication:BITRate <NR3>
TRIGger:A:COMMunication:BITRate?
```

Arguments <NR3> is a non-negative number greater than one and expressed as bits per second.

Examples TRIGGER:A:COMMUNICATION:BITRATE 1.544E6 sets the bit rate to 1.544 Mb/s.

TRIGGER:A:COMMUNICATION:BITRATE? might return :TRIGGER:A:COMMUNICATION:BITRATE 1.544e6, indicating a bit rate of 1.544 Mb/s.

TRIGger:A:COMMunication:CLOCk:POLarity

This command sets or queries the communication clock polarity

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Trigger

Syntax

```
TRIGger:A:COMMunication:CLOCk:POLarity {RISe|FALL}
TRIGger:A:COMMunication:CLOCk:POLarity?
```

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:COMMUNICATION:CLOCK:POLARITY RISE` sets RISE as the clock polarity.

`TRIGGER:A:COMMUNICATION:CLOCK:POLARITY?` might return `:TRIGGER:A:COMMUNICATION:CLOCK:POLARITY RISE`, indicating that the polarity is to trigger on the rising edge of the clock.

TRIGger:A:COMMunication:CMI:PULSEForm

This command sets or queries the CMI pulse form.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Trigger

Syntax

```
TRIGger:A:COMMunication:CMI:PULSEForm
{PLUSOne|MINUSOne|ZERO|EYEdiagram}
TRIGger:A:COMMunication:CMI:PULSEForm?
```

Arguments

`PLUSOne` triggers on a positive mark.

`MINUSOne` triggers on a negative mark.

`ZERO` triggers on the bit representing zero.

`EYEdiagram` positions the trigger point off screen to display the eye.

Examples

`TRIGGER:A:COMMUNICATION:CMI:PULSEFORM PLUSONE` sets triggering to occur on a positive mark.

`TRIGGER:A:COMMUNICATION:CMI:PULSEFORM?` might return `:TRIGGER:A:COMMUNICATION:CMI:PULSEFORM PLUSONE`, indicating that the pulse form is PLUSONE.

TRIGger:A:COMMunication:CODe

This command sets or queries the signal code that the communications trigger should expect on the incoming signal.

Conditions

This command is valid only if the instrument has Option MTM is installed.

Group

Trigger

Syntax

```
TRIGger:A:COMMunication:CODe
{AMI|HDB3|B3ZS|B6ZS|B8ZS|CMI|NRZ|MLT3|MANChester}
TRIGger:A:COMMunication:CODe?
```

Arguments

`AMI`

`HDB3`

`B3ZS`

`B6ZS`

`B8ZS`

`CMI`

`NRZ`

`MLT3`

`MANChester`

Examples

`TRIGGER:A:COMMUNICATION:CODE AMI` sets the AMI as the line code that the communications trigger expects on the incoming signal.

`TRIGGER:A:COMMUNICATION:CODE?` might return `:TRIGGER:A:COMMUNICATION:CODE AMI`, indicating that the code is AMI.

TRIGger:A:COMMunication:SOUrce

This command sets or queries the source channel.

Conditions This command is valid only if the instrument has Option MTM is installed.

Group Trigger

Syntax

```
TRIGger:A:COMMunication:SOUrce {CH1|CH2|CH3|CH4}
TRIGger:A:COMMunication:SOUrce?
```

Arguments

`CH1` argument selects CH 1 as the source channel.

`CH2` argument selects CH 2 as the source channel.

`CH3` argument selects CH 3 as the source channel.

`CH4` argument selects CH 4 as the source channel.

Examples

`TRIGGER:A:COMMUNICATION:SOURCE CH1` sets Channel 1 as the source channel.

`TRIGGER:A:COMMUNICATION:SOURCE?` might return `:TRIGGER:A:COMMUNICATION:SOURCE CH1`, indicating that the source is Channel 1.

TRIGger:A:COMMunication:SOUrce:TYPe

This command sets or queries the source type. This command works only when the Eye Diagram pulseform is selected.

Conditions This command is only valid if the instrument has Option MTM is installed.

Group Trigger

Syntax

```
TRIGger:A:COMMunication:SOUrce:TYPe {DATa|CLOCk|RECOVered}
TRIGger:A:COMMunication:SOUrce:TYPe?
```

Arguments

`DATa` causes the instrument to trigger and shift five unit intervals to form the expected eye pattern.

CLOCk causes the instrument to trigger but no shift occurs. The clock type causes random triggers with respect to the data channel, which must be one of the other three channels.

RECOVered causes the instrument to trigger on the recovered clock from the data signal attached to communication source forming an eye pattern on the source.

Examples

TRIGGER:A:COMMUNICATION:SOURCE:TYPE CLOCK sets the source type to clock.

TRIGGER:A:COMMUNICATION:SOURCE:TYPE? might return :TRIGGER:A:COMMUNICATION:SOURCE:TYPE DATA, indicating that the source type is data.

TRIGger:A:COMMunication:STANdard

This command sets or queries the standard that identifies the code and bit rate. The bit rate is used to compute the Unit Interval, which is the inverse of the bit rate. The Unit Interval influences time skew in an Eye Diagram, where you perform post processing on AMI isolated pulses, and pulse width settings if CMI.

Conditions

This command is valid only if the instrument has Option MTM is installed.

Group

Trigger

Syntax

```
TRIGger:A:COMMunication:STANdard
{ATAG1|ATAG2|ATAG3|CLOCKCoax|
CLOCKSymmetrical|Custom|D1|D2|DS0Contra|
DS0Double| DS0Single|DS0Timing|DS1|DS1A|
DS1C|DS2| DS2RATECoax|DS2RATESymmetrical|
DS3|DS4NA|E1|E2|E3|E4|ENET100|ENET1250|
ENETXAUI| FC133|FC266|FC531|FC1063|FC2125|FC4250|
FST|FW1394BS400B|FW1394BS1600B|HST|INF2_5G|
OC1|OC3| OC12|OC48|OC48_FEC|PCIEXPRESS|RATE32Mbit|
RATE97Mbit|RIO_500M|RIO_750M|
RIO_1G|RIO_2G|RIO_1_5G|RIO_SERIAL_1G|
RIO_SERIAL_2G|RIO_SERIAL_3G|SAS1_5|SAS3_?|
SFI5_2|SFI5_3|STM0_CMI|STM0_HDBX|STM1E|STS1| STS3|
TFI5_2|TFI5_3|VIDEO270|VIDEO292M|VIDEO360|VSROC192}
TRIGger:A:COMMunication:STANdard?
```

Related Commands

MASK:STANdard

Arguments

The following table shows various standards and their corresponding parameters.

Table 2-49: Communication Trigger Standards

Argument	Code	Bit Rate	Mask Type	Models
ATAG1	NRZ	1.5 Gb/s	Serial ATA	≥2.5 GHz
ATAG2	NRZ	3.0 Gb/s	Serial ATA	
ATAG3	NRZ	6.0 Gb/s	Serial ATA	
CLOCK Coax	None	2.048 Mb/s	ITU-T G.703	All
CLOCK Symmetrical	None	2.048 Mb/s	ITU-T G.703	All
Custom	AMI, B3ZS, B6ZS, B8ZS, CMI, HDB3, MLT3, NRZ	1.544 Mb/s		All
D1	NRZ	270 Mb/s	Video	All
D2	NRZ	143.18 Mb/s	Video	All
DS0 Contra	AMI	16.0 kb/s	ITU-T G.730	All
DS0 Double	AMI	32.0 kb/s	ITU-T G.730	All
DS0 Single	AMI	64.0 kb/s	ITU-T G.730	All
DS0 Timing	AMI	32.0 kb/s	ITU-T G.730	All
DS1	AMI B8ZS	1.544 Mb/s	ANSI T1.102	All
DS1A	AMI HDB3	2.048 Mb/s	ANSI T1.102	All
DS1C	AMI B8ZS	3.152 Mb/s	ANSI T1.102	All
DS2	AMI B6ZS	6.312 Mb/s	ANSI T1.102	All
DS2RATE Coax	AMI B8ZS	6.312 Mb/s	ITU-T G.703	All
DS2RATE Symmetrical	AMI B6ZS	6.312 Mb/s	ITU-T G.703	All
DS3	AMI B3ZS	44.736 Mb/s	ANSI T1.102	All
DS4NA	CMI	139.3 Mb/s	ANSI T1.102	All
E1	AMI HDB3	2.048 Mb/s	ITU-T G.703	All
E2	AMI HDB3	8.448 MB/s	ITU-T G.703	All
E3	AMI HDB3	34.368 Mb/s	ITU-T G.703	All
E4	CMI	139.26 Mb/s	ITU-T G.703	All
ENET100	MLT3	100 Mb/s	Ethernet	All
ENET1250	NRZ	1.25 Gb/s	Ethernet	≥2.5 GHz
ENETXAU	NRZ	3.125 Gb/s	Ethernet	≥
FC133	NRZ	132.8 Mb/s	Fibre Channel	All
FC266	NRZ	265.6 Mb/s	Fibre Channel	All
FC531	NRZ	531.2 Mb/s	Fibre Channel	≥1.0 GHz
FC1063	NRZ	1.0625 Gb/s	Fibre Channel	
FC2125	NRZ	2.125 Gb/s	Fibre Channel	
FC4250	NRZ	4.256 Gb/s	Fibre Channel	
FST	NRZ	12.0 Mb/s	USB 1.1/2.0	All
FW1394 BS400B	NRZ	491.5 Mb/s	IEEE1394b	≥1.0 GHz

Table 2-49: Communication Trigger Standards (cont.)

Argument	Code	Bit Rate	Mask Type	Models
FW1394 BS1600B	NRZ	1.966 Gb/s	IEEE1394b	
HST	NRZ	480 Mb/s	USB 1.1/2.0	≥1.0 GHz
INF_5G	NRZ	2.5 Gb/s	Infiniband	
OC1	NRZ	51.84 Mb/s	SONET/SDH	All
OC3	NRZ	155.52 Mb/s	SONET/SDH	All
OC12	NRZ	622.08 Mb/s	SONET/SDH	≥1.0 GHz
OC48	NRZ	2.4883 Gb/s	Sonet	
OC48 FEC	NRZ	2.666 Gb/s	Sonet	
PCIEXPRESS	NRZ	2.5 Gb/s	PCI-Express	
RATE32Mbit	NRZ	32.064 Mb/s	ITU	
RATE97Mbit	NRZ	97.728 Mb/s	ITU	
RIO_500M	NRZ	500 Mb/s	RapidIO/LP-LVDS	≥1.0 GHz
RIO_750_M	NRZ	1 Gb/s	RapidIO/LP-LVDS	
RIO_1G	NRZ	1 Gb/s	RapidIO/LP-LVDS	
RIO_1_5_G	NRZ	2 Gb/s	RapidIO/LP-LVDS	
RIO_SERIAL_1G	NRZ	1.25 Gb/s	RapidIO/LP-LVDS	
RIO_SERIAL_2G	NRZ	2.5 Gb/s	RapidIO/LP-LVDS	
RIO_SERIAL_3G	NRZ	3.125 Gb/s	RapidIO/LP-LVDS	
SAS1_5	NRZ		SAS	
SAS3_?	NRZ		SAS	
SFI5_2	NRZ	2.488 Gb/s	OIF	
SFI5_3	NRZ	3.125 Gb/s	OIF	
STM0_CMI	NRZ	51.84 Mb/s	ITU	
STM0_HDBX	NRZ	51.84 Mb/s	ITU	
STM1E	NRZ	155.52 Mb/s	ITU	
STS1	NRZ	51.840 Mb/s	ANSI T1.102	
STS3	NRZ	155.52 Mb/s	ANSI T1.102	
TFI5_2	NRZ	2.488 Gb/s	OIF	
TFI5_3	NRZ	3.1104 Gb/s	OIF	
VIDEO270	NRZ	270 Mb/s	Video	≥500 MHz
VIDEO292M	NRZ	1.485 Gb/s	Video	≥2.5 GHz
VIDEO360	NRZ	360 Mb/s	Video	≥500 MHz
VSROC192	NRZ	1.24116 Gb/s	OIF	

Examples `TRIGGER:A:COMMUNICATION:STANDARD E4` sets the standard to E4.

`TRIGGER:A:COMMUNICATION:STANDARD?` might return `:TRIGGER:A:COMMUNICATION:SOURCE:STANDARD DS1`.

TRIGger:A:EDGE:ENVelope

This command sets or queries whether the A edge trigger will use envelope triggering. DPO70000SX Series only.

Group Trigger

Syntax
```
TRIGger:A:EDGE:ENVelope <ON|OFF>
TRIGger:A:EDGE:ENVelope?
```

Arguments `OFF` disables edge envelope triggering.

`ON` enables edge envelope triggering.

Examples `TRIGGER:A:EDGE:ENVELOPE ON` enables edge envelope triggering.

`TRIGGER:A:EDGE:ENVELOPE?` may return ON or OFF.

TRIGger:A:HOLDoff? (Query Only)

NOTE. *Not available on some models.*

This query-only command 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. This command is equivalent to selecting Holdoff from the Trig menu and then viewing the current settings.

Group Trigger

Syntax
```
TRIGger:A:HOLDoff?
```

Related Commands TRIGger:A:HOLDoff:ACTUal?, TRIGger:A:HOLDoff:BY, 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 set to 900 ns.

TRIGger:A:HOLDoff:ACTUal? (Query Only)

NOTE. *Not available on some models.*

This query-only command returns the holdoff time actually used (expressed in seconds) by the A trigger. This command is equivalent to selecting Holdoff from the Trig menu and then viewing the current Trig Holdoff value.

Group Trigger

Syntax `TRIGger:A:HOLDoff:ACTUal?`

Related Commands TRIGger:A:HOLDoff?, TRIGger:A:HOLDoff:BY, TRIGger:A:HOLDoff:TIMe

Examples `TRIGGER:A:HOLDOFF:ACTUAL?` might return `:TRIGGER:A:HOLDOFF:ACTUAL 4.0000E-06`, indicating that the holdoff time is set to 4 µs.

TRIGger:A:HOLDoff:BY

This command sets or queries the type of holdoff for the A trigger. Holdoff types are expressed as either user-specified time (TIMe) or by an internally calculated minimum time value (DEFAult/AUTO). This command is equivalent to selecting Holdoff from the Trig menu and then setting the Holdoff type.

Group Trigger

Syntax
```
TRIGger:A:HOLDoff:BY {TIMe|DEFAult|RANDom|AUTO}
TRIGger:A:HOLDoff:BY?
```

Related Commands TRIGger:A:HOLDoff:TIMe

Arguments `TIMe` enables you to set the holdoff time via the TRIGger:A:HOLDoff:TIMe command.

DEFAult automatically calculates a holdoff time to use. This time is typically equivalent to the greater of 1/2 screen (5 divisions) of time or 250 ns. The maximum value is 12 s. For example, if the instrument is set to 1 ms/division then the default holdoff will be 1 ms/division x 25 divisions = 25 ms.

RANDom

AUTO

Examples

TRIGGER:A:HOLDOFF:BY TIME sets the holdoff to the "by time" setting. This enables you to set the holdoff time.

TRIGGER:A:HOLDOFF:BY? might return :TRIGGER:A:HOLDOFF:BY TIME, indicating that you will set the holdoff time.

TRIGger:A:HOLDoff:TIMe

This command sets or queries the A trigger holdoff time. This command is equivalent to selecting Holdoff from the Trig menu and then choosing the desired Trig Holdoff.

Group

Trigger

Syntax

```
TRIGger:A:HOLDoff:TIMe <NR3>
TRIGger:A:HOLDoff:TIMe?
```

Related Commands

TRIGger:A:HOLDoff:BY

Arguments

<NR3> specifies the holdoff time in seconds. The range is from 250 ns through 12.0 s.

Examples

TRIGGER:A:HOLDOFF:TIME 10 sets the A trigger holdoff time to 10 s.

TRIGGER:A:HOLDOFF:TIME? might return :TRIGGER:A:HOLDOFFTIME 1.2000E-06, indicating that the A trigger holdoff time is set to 1.2 µs.

TRIGger:A:I2C:ADDRess:MODe

DPO7000 Series only: This command sets or queries the I2C address mode to 7 or 10 bit.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:I2C:ADDRess:MODe {ADDR7|ADDR10}
TRIGger:A:I2C:ADDRess:MODe?
```

Arguments `ADDR7` specifies the 7-bit I2C address mode.

`ADDR10` specifies the 10-bit I2C address mode.

Examples `TRIGGER:A:I2C:ADDRESS:MODE ADDR10` sets the I2C address mode to 10 bit.

`TRIGGER:A:I2C:ADDRESS:MODE?` might return `:TRIGGER:A:I2C:ADDRESS:MODE ADD7`, indicating that the I2C address mode is set to 7 bit.

TRIGger:A:I2C:ADDRess:RWINClude

Sets the I2C read/write bit to address type.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:I2C:ADDRess:RWINClude {OFF|ON}
TRIGger:A:I2C:ADDRess:RWINClude?
```

Arguments `ON` specifies the I2C read/write bit to address type.

`OFF` specifies to stop the I2C read/write bit to address type.

Examples `TRIGGER:A:I2C:ADDRESS:RWINCLUDE ON` sets the read/write bit to address type.

`TRIGGER:A:I2C:ADDRESS:RWINCLUDE?` might return `:TRIGGER:A:I2C:ADDRESS:RWINCLUDE OFF`, indicating that the read/write bit stops specifying the address type.

TRIGger:A:I2C:ADDRess:TYPe

DPO7000 Series only: This command sets or queries the I2C address type.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:I2C:ADDRess:TYPe
{GENeralcall|STARtbyte|TENbit|NONe}
TRIGger:A:I2C:ADDRess:TYPe?
```

Arguments

`GENERALCALL` specifies a generalcall address type.

`STARTBYTE` specifies a startbyte address type.

`TENBIT` specifies a tenbit address type.

`NONE` specifies I2C address type of none.

Examples

`TRIGGER:A:I2C:ADDRESS:TYPE NONE` sets the I2C address type to none.

`TRIGGER:A:I2C:ADDRESS:TYPE?` might return `:TRIGGER:A:I2C:ADDRESS:TYPE TENBIT`, indicating that the I2C address type is set to tenbit.

TRIGger:A:I2C:ADDRess:VALue

DPO7000 Series only: This command sets or queries the binary address string used for the I2C trigger if the trigger condition is ADDR or ADDRANDDATA.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:I2C:ADDRess:VALue <QString>
TRIGger:A:I2C:ADDRess:VALue?
```

Arguments

`<QString>` is up to 7 bits or 10 bits depending on the address mode that specifies the address.

Examples

`TRIGGER:A:I2C:ADDRESS:VALUE "1011"` sets the I2C address value to 1011.

`TRIGGER:A:I2C:ADDRESS:VALUE?` might return `:TRIGGER:A:I2C:ADDRESS:VALUE "10111"`, indicating that the I2C address value is set to 10111.

TRIGger:A:I2C:CLOCk:LEVel

DPO7000 Series only: This command sets or queries the clock source for the I2C Serial Trigger.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:I2C:CLOCk:LEVel <NR3>
TRIGger:A:I2C:CLOCk:LEVel?
```

Arguments <NR3> specifies the I2C clock level. The clock level range is ±5.

Examples TRIGGER:A:I2C:CLOCK:LEVEL0.0 sets the I2C clock level to 0.0.

TRIGGER:A:I2C:CLOCK:LEVEL? might return :TRIGGER:A:I2C:CLOCK:LEVEL 5.0000, indicating that the I2C clock level is set to 5.

TRIGger:A:I2C:CLOCk:SOUrce

DPO7000 Series only: This command sets or queries the clock source for the I2C Serial Trigger.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:I2C:CLOCk:SOUrce CH<x>
TRIGger:A:I2C:CLOCk:SOUrce?
```

Arguments CH<x> specifies the clock source. x can be 1 through 4.

Examples TRIGGER:A:I2C:CLOCK:SOURCE CH1 sets the I2C clock source to channel CH1.

TRIGGER:A:I2C:CLOCK:SOURCE? might return :TRIGGER:A:I2C:CLOCK:SOURCE CH3, indicating that the I2C clock source is set to CH3.

TRIGger:A:I2C:CONDition

DPO7000 Series only: This command sets or queries the trigger condition for the I2C trigger.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:I2C:CONDition
{STARt|STOP|REPEATstart|ACKMISS|ADDRess|DATA|ADDRANDDATA}
TRIGger:A:I2C:CONDition?
```

Arguments

`START` specifies triggering on the start of a packet.

`STOP` specifies triggering on the stop packet.

`REPEATstart` specifies triggering on a repeated start of packet.

`ACKMISS` specifies triggering on a missing acknowledgement.

`ADDRess` specifies triggering on the address.

`DATA` specifies triggering on the specified data.

`ADDRANDDATA` specifies triggering on the specified data and address.

Examples

`TRIGGER:A:I2C:CONDITION START` sets the trigger condition for the I2C trigger on the start of the packet.

`TRIGGER:A:I2C:CONDITION?` might return `:TRIGGER:A:I2C:CONDITION DATA`, indicating that the trigger condition for the I2C trigger is set on specified data.

TRIGger:A:I2C:DATa:DIRection

DPO7000 Series only: This command sets or queries the I2C trigger condition valid on a READ, WRITE, or either.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:I2C:DATa:DIRection {READ|WRITE|NOCARE}
TRIGger:A:I2C:DATa:DIRection?
```

Arguments

READ specifies read as the data direction.

WRITE specifies write as the data direction.

NOCARE specifies either as the data direction.

Examples

TRIGGER:A:I2C:DATA:DIRECTION WRITE sets the I2C data direction to write.

TRIGGER:A:I2C:DATA:DIRECTION? might return :TRIGGER:A:I2C:DATA:DIRECTION READ, indicating that the I2C data direction is set on read.

TRIGger:A:I2C:DATa:LEVel

DPO7000 Series only: This command sets or queries the threshold level for the I2C data source.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:I2C:DATa:LEVel <NR3>
TRIGger:A:I2C:DATa:LEVel?
```

Arguments

<NR3> specifies the I2C data level with a range of ±%.0.

Examples

TRIGGER:A:I2C:DATA:LEVEL 0.0 sets the I2C data level to 0.0.

TRIGGER:A:I2C:DATA:LEVEL? might return :TRIGGER:A:I2C:DATA:LEVEL 0.0, indicating that the I2C data level is set on 0.0.

TRIGger:A:I2C:DATa:SOUrce

DPO7000 Series only: This command sets or queries the data source for the I2C serial trigger.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:I2C:DATa:SOUrce CH<x>
TRIGger:A:I2C:DATa:SOUrce?
```

Arguments CH<x> specifies the data source. x can be 1 through 4.

Examples TRIGGER:A:I2C:DATA:SOURCE CH1 sets the data source for the I2C serial trigger to channel CH1.

TRIGGER:A:I2C:DATA:SOURCE? might return :TRIGGER:A:I2C:DATA:SOURCE CH4, indicating that the data source for the I2C serial trigger is set on channel CH4.

TRIGger:A:I2C:DATa:STARt

DPO7000 Series only: This command sets or queries the data start byte. This byte is only needed if triggering on data. This byte is the first byte that pattern matching uses. The byte is a destination offset from the start of the packet.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:I2C:DATa:STARt <NR3>
TRIGger:A:I2C:DATa:STARt?
```

Arguments <NR3> specifies the data start byte (offset), with a range of 0 to 65535.

Examples TRIGGER:A:I2C:DATA:START 0 specifies the I2C data start byte offset to 0.

TRIGGER:A:I2C:DATA:START? might return :TRIGGER:A:I2C:DATA:START 5, indicating that the I2C data start byte offset is set to 5.

TRIGger:A:I2C:DATa:VALue

DPO7000 Series only: This command sets or queries the binary data string used for I2C triggering if the trigger condition is DATA or ADDRANDDATA.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:I2C:DATa:VALue <QString>
TRIGger:A:I2C:DATa:VALue?
```

Arguments <QString> sets the I2C data value. This value can be 1 to 32 bits.

Examples TRIGGER:A:I2C:DATA:VALUE "1011" sets the I2C data value to 1011.

TRIGGER:A:I2C:DATA:VALUE? might return :TRIGGER:A:I2C:DATA:VALUE "10111", indicating that the I2C data value is set to 10111.

TRIGger:A:I2C:FORMat

DPO7000 Series only: This command sets or queries the display format for the I2C data value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:I2C:FORMat {BINary|HEX}
TRIGger:A:I2C:FORMat?
```

Arguments BINary specifies binary format for I2C triggering.

HEX specifies hexadecimal format for I2C triggering.

Examples TRIGGER:A:I2C:FORMAT HEX sets the I2C trigger format to hexadecimal.

TRIGGER:A:I2C:FORMAT? might return :TRIGGER:A:I2C:FORMAT BINARY, indicating that the I2C trigger format is set to binary.

TRIGger:A:MODe

This command sets or queries the A trigger mode. This command is equivalent to selecting Mode from the Trig menu and then choosing the desired Trigger Mode.

Group Trigger

Syntax

```
TRIGger:A:MODe {AUTO|NORMal}
TRIGger:A:MODe?
```

Related Commands TRIGger:{A|B}:LEVel

Arguments AUTO generates a trigger if one is not detected within a specified time period.

NORMal waits for a valid trigger event.

Examples TRIGGER:A:MODE NORMAL sets the trigger mode to normal.

TRIGGER:A:MODE? might return :TRIGGER:A:MODE AUTO, indicating that the trigger mode is auto.

TRIGger:A:PULse:GLItch:ENVelope

This command sets or queries whether the A glitch trigger will use envelope triggering. DPO70000SX Series only.

Group Trigger

Syntax

```
TRIGger:A:PULse:GLItch:ENVelope <OFF|ON>
TRIGger:A:PULse:GLItch:ENVelope?
```

Arguments OFF disables glitch envelope triggering.

ON enables glitch envelope triggering.

Examples TRIGGER:A:PULSE:GLITCH:ENVELOPE ON enables glitch envelope triggering.

TRIGGER:A:PULSE:GLITCH:ENVELOPE? may return ON or OFF.

TRIGger:A:PULse:PERiod? (Query Only)

This query-only command returns the Pulse Trigger settings specific to Period/Frequency trigger. These include the array of high and low limits, the “trigger when” enumeration, the polarity enumeration, the qualification enumeration, and the view enumeration. The view enumeration controls whether the high and low limits are expressed in units of time (Seconds) or frequency (Hz). The qualification enumeration determines if the trigger is state-qualified by additional signals (channels). The polarity enumeration determines whether the rising or falling edges of the signal are used to initiate the trigger. The

trigger when enumeration determines if the trigger should occur when the period (frequency) is greater than or less than a single (low) limit value, or within or outside the period (frequency) of a high-low limit period (frequency) pair.

The Period/Frequency Trigger is similar to Width Trigger, but instead of triggering on a single pulse of a given width, it triggers on the period (two back-to-back pulses, one up and one down) of a given width. As a convenience the period limits (Seconds) can be expressed as frequency limits (Hz) by changing the view enumeration. It is often used to determine if oscillators are operating within specification.

Group Trigger

Syntax

```
TRIGger:A:PULse:PERiod?
```

Related Commands TRIGger:A:PULse:PERiod?TRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:POLarityTRIGger:A:PULse:PERiod:QUAlifyTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod?TRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:WHEnTRIGger:B:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:QUAlifyTRIGger:B:PULse:PERiod:VIEW

Examples `TRIGGER:A:PULSE:PERIOD?` might return `:TRIGGER:A:PULSE:PERIOD: '90.0000E-9;10.0000;GREATERTHAN;POSITIVE;OCCURS\n'`, indicating the period limits, when enumeration, polarity, and so forth.

TRIGger:A:PULse:PERiod:HIGHLimit

This command sets or queries the Period/Frequency high limit. When the VIEW is PERiod the units of this number are time (Seconds). When the VIEW is FREQuency, the units of this number are frequency (Hz). The high limit is used in combination with the low limit to define a range if the WHEn enumeration is WITHin or OUTside. These limits specify the most significant characteristic of the signal for this Pulse Trigger Type.

Group Trigger

Syntax

```
TRIGger:A:PULse:PERiod:HIGHLimit <NR3>
TRIGger:A:PULse:PERiod:HIGHLimit?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:

POLarityTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:WHEnTRIGger:B:PULse:PERiod:POLarity

Arguments <NR3> is the period (in Seconds) or frequency (in Hz).

Examples TRIGGER:A:PULSE:PERIOD:HIGHLIMIT 101.0e-9 sets the Pulse Period Trigger High Limit to one hundred one nanoseconds, which, in combination with a Low Limit of 99.0e-9, is suitable for triggering on a 10MHz clock if the WHEn enumeration is set to WITHin.

TRIGGER:A:PULSE:PERIOD:HIGHLIMIT? might return :TRIGGER:A:PULSE:PERIOD:HIGHLIMIT 101.0e-9 (if View is Period), or :TRIGGER:A:PULSE:PERIOD:HIGHLIMIT 10.1e6 (if View is Frequency).

TRIGger:A:PULse:PERiod:LOWLimit

This command sets or queries the Period/Frequency low limit. When the VIEW is PERiod the units of this number are time (Seconds). When the VIEW is FREQuency, the units of this number are frequency (Hz). The low limit is used alone when the WHEn enumeration is LESSthan or GREATerthan, and is used in combination with the high limit to define a range if the WHEn enumeration is WITHin or OUTside. The limits specify the most significant characteristic of the signal for this Pulse Trigger Type.

Group Trigger

Syntax
```
TRIGger:A:PULse:PERiod:LOWLimit <NR3>
TRIGger:A:PULse:PERiod:LOWLimit?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:WHEnTRIGger:B:PULse:PERiod:POLarity

Arguments <NR3> is the period (in Seconds) or frequency (in Hz).

Examples TRIGGER:A:PULSE:PERIOD:LOWLIMIT 101.0e-9 sets the Pulse Period Trigger Low Limit to one hundred one nanoseconds, which is suitable for triggering on a 10 MHz clock if the WHEn enumeration is set to LESSthan.

TRIGGER:A:PULSE:PERIOD:LOWLIMIT? might return :TRIGGER:A:PULSE:PERIOD:LOWLIMIT 101.0e-9 (if View is Period), or might return :TRIGGER:A:PULSE:PERIOD:LOWLIMIT 101.0e-9 (if View is Period), or :TRIGGER:A:PULSE:PERIOD:LOWLIMIT 9900990. (if View is Frequency), (if View is Frequency).

TRIGger:A:PULse:PERiod:POLarity

This command sets or queries the polarity for the Pulse Period trigger. It refers to the polarity of the edges that begin and end a given period of the signal, and thus allows duty-cycle testing of clock signals.

Group Trigger

Syntax

```
TRIGger:A:PULse:PERiod:POLarity {NEGAtive|POSITIVe}
TRIGger:A:PULse:PERiod:POLarity?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:POLarityTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:VIEW

Arguments POSITIVe specifies to trigger on the rising or positive edge of a signal.

NEGAtive specifies to trigger on the falling or negative edge of a signal.

Examples TRIGGER:A:PULSE:PERIOD:POLARITY POSITIVE sets the A pulse period trigger slope to positive, which triggers on the rising edge of the signal.

TRIGGER:A:PULSE:PERIOD:POLARITY? might return :TRIGGER:A:PULSEL:PERIOD:POLARITY NEGATIVE, indicating that the A pulse period trigger polarity is negative.

TRIGger:A:PULse:PERiod:QUALify

This command sets or queries the Qualification setup for Pulse Period Trigger. The high/low state of one or more other signals (channels) may be used to qualify whether the trigger should occur or not. The BUS option is not available on the SX series instruments at this time.

When the QUALify enumeration is OCCurs, no qualification is done.

When the QUALify enumeration is LOGIC, the scope triggers using the pulse period trigger definition as qualified by the high/low states of the qualifying signals. Additional description is given elsewhere in this document under the term “logic qualification.”

Group Trigger

Syntax

```
TRIGger:A:PULse:PERiod:QUAlify {OCCurs | LOGIC | BUS}
TRIGger:A:PULse:PERiod:QUAlify?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:WHEnTRIGger:B:PULse:PERiod:VIEW

Arguments

`OCCurs` specifies the no other signals are used to qualify the pulse period trigger.

`LOGIC` specifies that other analog signals (channels) are used to qualify the pulse period trigger.

`BUS` specifies that other digital signals (digital channels) are used to qualify the pulse period trigger.

Examples

`TRIGGER:A:PULSE:PERIOD:QUALIFY LOGIC` specifies that one or more other analog signals (channels) are used to qualify the pulse period trigger.

`TRIGGER:A:PULSE:PERIOD:QUALIFY?` might return `:TRIGGER:A:PULSE:PERIOD:QUALIFY OCCURS`, indicating that the A pulse period trigger is not qualified by any other signals (channels).

TRIGger:A:PULse:PERiod:VIEW

This command sets or queries the “view” for the pulse period trigger. When PERiod is selected the LOWLimit and HIGHLimit values are in units of time (Seconds). When FREQuency is selected, the units are in frequency (Hz). This two methods are equivalent, so the alternative views are provided as a convenience.

Group Trigger

Syntax

```
TRIGger:A:PULse:PERiod:VIEW {PERiod|FREQuency}
TRIGger:A:PULse:PERiod:VIEW?
```

Related Commands TRIGger:{A|B}:EDGE:SOUrceTRIGger:{A|B}: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:PULSE:PERIOD:VIEW PERIOD` sets the interpretation of the high and low limits to units of time (Seconds).

`TRIGGER:A:PULSE:PERIOD:VIEW?` might return `:TRIGGER:A:PULSE:PERIOD:VIEW FREQUENCY`, indicating that the interpretation of the high and low limits is in units of frequency (Hz).

TRIGger:A:PULse:PERiod:WHEn

This command sets or queries the "trigger when" enumeration for the Pulse Period Trigger.

If the argument is LESSthan, the scope triggers on the signal when the period (frequency) of the signal is less than the period (frequency) of the Low Limit.

If the argument is GREATerthan, the scope triggers on the signal when the period (frequency) of the signal is greater than the period (frequency) of the Low Limit.

If the argument is WITHin, the scope triggers on the signal when the period (frequency) of the signal is within the range defined by the Low Limit and the High Limit.

If the argument is OUTside, the scope triggers on the signal when the period (frequency) of the signal is outside of the range defined by the Low Limit and the High Limit.

Group Trigger

Syntax

```
TRIGger:A:PULse:PERiod:WHEn {LESSthan | GREATerthan | WITHin
| OUTside}
TRIGger:A:PULse:PERiod:WHEn?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:POLarityTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:VIEW

Arguments

LESSthan tells the scope to trigger when the period (frequency) of the signal is less than the low limit value.

GREATerthan tells the scope to trigger when the period (frequency) of the signal is greater than the low limit value.

WITHin tells the scope to trigger when the period (frequency) of the signal is inside the range set by the low and high limit values.

OUTside tells the scope to trigger when the period (frequency) of the signal is outside of the range set by the low and high limit values.

Examples

TRIGGER:A:PULSE:PERIOD:WHEN LESSTHAN tells the scope to trigger when the period (frequency) of the signal is less than the low limit value.

TRIGGER:A:PULSE:PERIOD:WHEN WITHin tells the scope to trigger when the period (frequency) of the signal is inside the range set by the low and high limit values.

TRIGGER:A:PULSE:PERIOD:WHEN? might return GREATERTHAN indicating the instrument will trigger when the period (frequency) of the signal is less than the low limit value.

TRIGger:A:PULse:TIMEOut:ENVelope

This command sets or queries whether the A timeout trigger will use envelope triggering. DPO70000SX Series only.

Group

Trigger

Syntax

```
TRIGger:A:PULse:TIMEOut:ENVelope <OFF|ON>
TRIGger:A:PULse:TIMEOut:ENVelope?
```

Arguments

OFF disables timeout envelope triggering.

ON enables timeout envelope triggering.

Examples

TRIGGER:A:PULSE:TIMEOUT:ENVELOPE ON enables timeout envelope triggering.

TRIGGER:A:PULSE:TIMEOUT:ENVELOPE? may return ON or OFF.

TRIGger:A:PULse:WIDth:ENVelope

This command sets or queries whether the A width trigger will use envelope triggering. DPO70000SX Series only.

Group Trigger

Syntax

```
TRIGger:A:PULse:WIDth:ENVelope <OFF|ON>
TRIGger:A:PULse:WIDth:ENVelope?
```

Arguments `OFF` disables width envelope triggering.

`ON` enables width envelope triggering.

Examples `TRIGGER:A:PULSE:WIDTH:ENVELOPE OFF` disables width envelope triggering.

`TRIGGER:A:PULSE:WIDTH:ENVELOPE?` may return ON or OFF.

TRIGger:A:RS232:BAUd

DPO7000 Series only: This command sets or queries the baud rate for RS232 triggering.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:RS232:BAUd <x>
TRIGger:A:RS232:BAUd?
```

Arguments <x> specifies the baud rate for RS232 triggering. X can be: 1500000, 921600. 750000, 460800, 115200, 57600, 38400, 19200, 9600, 7200, 4800, 2400, 2000, 1800, 1200, 600, 300, 150, or 75.

Examples `TRIGGER:A:RS232:BAUD 2000` sets the RS232 triggering baud rate to 2000.

`TRIGGER:A:RS232:BAUD?` might return `:TRIGGER:A:RS232:BAUD 1800`, indicating that the RS232 triggering baud rate is set to 1800.

TRIGger:A:RS232:DATa:LEVel

DPO7000 Series only: This command sets or queries the threshold for the RS232 data source.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:RS232:DATa:LEVel <NR3>
TRIGger:A:RS232:DATa:LEVel?
```

Arguments <NR3> specifies the threshold level with a range of ±5.0.

Examples TRIGGER:A:RS232:DATA:LEVEL 0.0 sets the threshold for the RS232 data source to 0.0.

TRIGGER:A:RS232:DATA:LEVEL? might return :TRIGGER:A:RS232:DATA:LEVEL 5.0, indicating that the threshold for the RS232 data source is set to 5.0.

TRIGger:A:RS232:DATa:SOUrce

DPO7000 Series only: This command sets or queries the RS232 data source.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:RS232:DATa:SOUrce CH<x>
TRIGger:A:RS232:DATa:SOUrce?
```

Arguments CH<x> specifies the RS232 data source. x can be 1, 2, 3, or 4.

Examples TRIGGER:A:RS232:DATA:SOURCE CH2 sets the RS232 data source to channel CH2.

TRIGGER:A:RS232:DATA:SOURCE? might return :TRIGGER:A:RS232:DATA:SOURCE CH1, indicating that the RS232 data source is set to channel CH1.

TRIGger:A:RS232:DATa:VALue

DPO7000 Series only: This command sets or queries the binary data string used for RS232 triggering.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:RS232:DATa:VALue <QString>
TRIGger:A:RS232:DATa:VALue?
```

Arguments <QString> specifies the RS232 data value. The value can be up to 8 bits.

Examples TRIGGER:A:RS232:DATA:VALUE "1011" sets the RS232 triggering data value to 1011.

TRIGGER:A:RS232:DATA:VALUE? might return :TRIGGER:A:RS232:DATA:VALUE "10111", indicating that the RS232 data value is set to 10111.

TRIGger:A:RS232:FORMat

DPO7000 Series only: This command sets or queries the display format for the RS232 data value.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:RS232:FORMat {BINary|HEX}
TRIGger:A:RS232:FORMat?
```

Arguments BINary specifies binary for the display format.

HEX specifies hexadecimal for the display format.

Examples TRIGGER:A:RS232:FORMAT HEX sets the display format for the RS232 data to hexadecimal.

TRIGGER:A:RS232:FORMAT? might return :TRIGGER:A:RS232:DATA:FORMAT BINARY, indicating that the display format for the RS232 data is set to binary.

TRIGger:A:RS232:PARity

DPO7000 Series only: This command sets or queries the state of the RS232 parity bit.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:RS232:PARity {NONe|EVEN|ODD}
TRIGger:A:RS232:PARity?
```

Arguments

NONe specifies no parity.

EVEN specifies even RS232 parity.

ODD specifies odd RS232 parity.

Examples

TRIGGER:A:RS232:PARITYEVEN sets the RS232 parity to even.

TRIGGER:A:RS232:PARITY? might return :TRIGGER:A:RS232:PARITY ODD, indicating that the RS232 parity is set to odd.

TRIGger:A:SERIAL:BITRate

This command sets or queries the clock/data bit rate. Changing the bit rate causes the standard to become “custom”. It remains “custom” until another standard is chosen.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:BITRate <NR3>
TRIGger:A:SERIAL:BITRate?
```

Arguments

<NR3> This is the series bit rate and is expressed in bits per second. The range is 1.5e6 to 1.25e9.

Examples

`TRIGGER:A:SERIAL:BITRATE 51.8400E+6` sets the bit rate to 51.84 Mbits/second.

`TRIGGER:A:SERIAL:BITRATE?` might return `:TRIGGER:A:SERIAL:BITRATE 51.8400E+6`, indicating that the bit rate is set to 51.84 Mbits/second.

TRIGger:A:SERIAL:CLOCk:LEVel

This command sets or queries the serial trigger clock level.

Group

Trigger

Syntax

```
TRIGger:A:SERIAL:CLOCk:LEVel <NR3>
TRIGger:A:SERIAL:CLOCk:LEVel?
```

Arguments

`<NR3>` is a value in the range of 9.9E37 and is expressed in volts.

Examples

`TRIGGER:A:SERIAL:CLOCK:LEVEL 0.0` sets the clock level to 0 V.

`TRIGGER:A:SERIAL:CLOCK:LEVEL?` might return `:TRIGGER:A:SERIAL:CLOCK:LEVEL 51.8400E+6`, indicating that the 32.0000E-3 indicating that the serial trigger clock level is set to 51.84E6 volts.

TRIGger:A:SERIAL:CLOCk:POLarity

This command sets or queries the serial clock polarity.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:CLOCk:POLarity {RISe|FALL}
TRIGger:A:SERIAL:CLOCk:POLarity?
```

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:SERIAL:CLOCK:POLARITY RISE` sets the clock polarity to rise.

`TRIGGER:A:SERIAL:CLOCK:POLARITY?` might return `:TRIGGER:A:SERIAL:CLOCK:POLARITY FALL`, indicating that the clock polarity is set to fall.

TRIGger:A:SERIAL:CLOCk:SOUrce

This command sets or queries the serial data source channel.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:CLOCk:SOUrce {CH<x>|RECOVered}
TRIGger:A:SERIAL:CLOCk:SOUrce?
```

Arguments

`CH<x>` specifies one of the input channels. x can be 1, 2, 3, or 4.

`RECOVered` specifies clock polarity and level are extracted from the serial data stream and other clock parameters are ignored.

Examples

`TRIGGER:A:SERIAL:CLOCK:SOURCE CH1` sets the serial source channel to CH1.

`TRIGGER:A:SERIAL:CLOCK:SOURCE?` might return `:TRIGGER:A:SERIAL:CLOCK:SOURCE CH3`, indicating that the serial source channel is set to CH3.

TRIGger:A:SERIAL:CODe

This command sets or queries the signal code.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:CODe {NRZ|S8B10B}
TRIGger:A:SERIAL:CODe?
```

Arguments

`NRZ` sets the code to NRZ.

`S8B10B` sets the code to S8B0B.

Examples

`TRIGGER:A:SERIAL:CODE NRZ` sets the signal code to NRZ.

`TRIGGER:A:SERIAL:CODE?` might return `:TRIGGER:A:SERIAL:CODE S8B10B`, indicating that the signal code is set to S8B0B.

TRIGger:A:SERIAL:DATa:FORMat

This command sets or queries how the Pattern string is formatted.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:DATa:FORMat {BINary|HEX}
TRIGger:A:SERIAL:DATa:FORMat?
```

Arguments

`BINary` pattern symbols are: {‘SPACE’|0|1|X} ‘SPACE’ is white space and can be included to make the 32-bit pattern easier to read when setting the pattern.

`HEX` pattern symbols are used only with NRZ. They are: {‘ ‘|0|1|2|3|4|5|6|7|8|9|A\B|C|D\E|F|X|?} Hex ‘X” represents the bit pattern ‘XXXX’. Hex ‘?’ represents any other binary bit pattern which doesn’t have a hex representation, such as ‘00X1’.

Examples

`TRIGGER:A:SERIAL:DATA:FORMAT BINARY` sets the Pattern string to a binary format.

`TRIGGER:A:SERIAL:DATA:FORMAT?` might return `:TRIGGER:A:SERIAL:CODE HEX`, indicating that the Pattern string is set to a hexadecimal format.

TRIGger:A:SERIAL:DATa:PATtern

This command sets or queries the data pattern to allow up to 64-bit serial patterns.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:DATa:PATtern <QString>
TRIGger:A:SERIAL:DATa:PATtern?
```

Related Commands TRIGger:A:SERIAL:DATa:PATtern:NRZ

TRIGger:A:SERIAL:DATa:PATtern:S8B10B

Arguments <QString> specifies the serial pattern to trigger on. The default value is “01”, such that there are up to 64 characters total.

Examples If one sets the data format to binary with a TRIGGER: A:SERIAL:DATA:FORMAT command, as follows: TRIGger:A:SERIAL:DATA:FORMAT BIN then the following query: TRIGger:A:SERIAL:DATA:PATTERN? might return the following characters: 1101XXXX10.

If one next sets the data format to hexadecimal with the TRIGGER:A:SERIAL:DATA:FORMAT commands as follows: TRIGger:A:SERIAL:DATA:FORMAT HEX then the following query: TRIGger:A:SERIAL:DATA:PATTERN? might return the following characters: 3?? Showing, in this case, 3 followed by two “?”. Each hex character has replaced four binary characters. The hex “??” has replaced the binary ”01XXXX10”. The “?” was used because "01XX” and “XX10” do not have exact hex representations.

If one sends: TRIGger:A:SERIAL:DATA:PATTERN “XXXX XXXa” this will set the pattern to eight hex digits. The space in the middle only serves to make the pattern easier to read.

Now, if one sends the following query: TRIGger:A:SERIAL:DATA:PATTERN? this might return: XXXXXXXA showing the effect of the pattern sent in example 3. The cosmetic space from example 3 is not returned.

Finally, one could change the data format back to binary with the TRIGGER:A:SERIAL:DATA:FORMAT command, as follows: TRIGGER:A:SERIAL:DATA:FORMAT BIN then one could query the instrument using the pattern command, as follows: TRIGGER:A:SERIAL:DATA:PATTERN? This could return: XXXXXXXXXXXXXXXXXXXXXXXXXXXX1010 showing the response due to the binary format. Four binary characters “1010” now replace the hex character “A”.

TRIGger:A:SERIAL:DATa:PATtern:NRZ

This command sets or queries the data pattern to allow up to 64-bit serial patterns.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:DATa:PATtern:NRZ <QString>
TRIGger:A:SERIAL:DATa:PATtern:NRZ?
```

Related Commands TRIGger:A:SERIAL:DATa:PATtern

TRIGger:A:SERIAL:DATa:PATtern:S8B10B

Arguments <QString> specifies the serial pattern to trigger on. The default value is “01”, such that there are up to 64 characters total.

Examples If one sets the data format to binary with a TRIGGER: A:SERIAL:DATA:FORMAT command, as follows: TRIGger:A:SERIAL:DATA:FORMAT BIN then the following query: TRIGGER:A:SERIAL:DATA:PATTERN:NRZ? might return the following characters: 1101XXXX10.

If one next sets the data format to hexadecimal with the TRIGGER:A:SERIAL:DATA:FORMAT commands as follows: TRIGger:A:SERIAL:DATA:FORMAT HEX then the following query: TRIGGER:A:SERIAL:DATA:PATTERN:NRZ? might return the following characters: 3?? Showing, in this case, 3 followed by two “?”. Each hex character has replaced four binary characters. The hex “??” has replaced the binary ”01XXXX10”. The “?” was used because "01XX” and “XX10” don’t have an exact hex representations.

If one sends: TRIGger:A:SERIAL:DATA:PATTERN:NRZ “XXXX XXXa” this will set the pattern to eight hex digits. The space in the middle only serves to make the pattern easier to read.

Now, if one sends the following query: TRIGGER:A:SERIAL:DATA:PATTERN:NRZ? this might return: XXXXXXXA showing the effect of the pattern sent in example 3. The cosmetic space from example 3 is not returned.

Finally, one could change the data format back to binary with the TRIGGER:A:SERIAL:DATA:FORMAT command, as follows: TRIGGER:A:SERIAL:DATA:FORMAT BIN then one could query the instrument using the pattern command, as follows: TRIGGER:A:SERIAL:DATA:PATTERN:NRZ? This could return: XXXXXXXXXXXXXXXXXXXXXXXXXXXX1010 showing the response due to the binary format. Four binary characters “1010” now replace the hex character “A”.

TRIGger:A:SERIAL:DATa:PATtern:S8B10B

NOTE. *Not available on some models.*

This command sets or queries the data pattern to allow up to 64-bit serial patterns.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:DATa:PATtern:S8B10B <QString>
TRIGger:A:SERIAL:DATa:PATtern:S8B10B?
```

Related Commands TRIGger:A:SERIAL:DATa:PATtern

TRIGger:A:SERIAL:DATa:PATtern:NRZ

Arguments `<QString>` specifies the serial pattern to trigger on. The default value is "0011111010", such that there are up to 64 characters total.

Examples If one sets the data format to binary with a `TRIGGER: A:SERIAL:DATA:FORMAT` command, as follows: `TRIGger:A:SERIAL:DATA:FORMAT BIN` then the following query: `TRIGGER:A:SERIAL:DATA:PATTERN:S8B10B?` might return the following characters: `1101XXXX10`.

If one next sets the data format to hexadecimal with the `TRIGGER:A:SERIAL:DATA:FORMAT` commands as follows: `TRIGger:A:SERIAL:DATA:FORMAT HEX` then the following query: `TRIGGER:A:SERIAL:DATA:PATTERN:S8B10B?` might return the following characters: `3??` Showing, in this case, 3 followed by two "?". Each hex character has replaced four binary characters. The hex "??" has replaced the binary "01XXXX10". The "?" was used because "01XX" and "XX10" don't have an exact hex representations.

If one sends: `TRIGGER:A:SERIAL:DATA:PATTERN:S8B10B "XXXX XXXa"` this will set the pattern to eight hex digits. The space in the middle only serves to make the pattern easier to read.

Now, if one sends the following query: `TRIGGER:A:SERIAL:DATA:PATTERN:S8B10B?` this might return: `XXXXXXXA` showing the effect of the pattern sent in example 3. The cosmetic space from example 3 is not returned.

Finally, one could change the data format back to binary with the `TRIGGER:A:SERIAL:DATA:FORMAT` command, as follows: `TRIGGER:A:SERIAL:DATA:FORMAT BIN` then one could query the instrument using the pattern command, as follows:

TRIGGER:A:SERIAL:DATA:PATTERN:S8B10B? This could return: XXXXXXXXXXXXXXXXXXXXXXXXXXXX1010 showing the response due to the binary format. Four binary characters “1010” now replace the hex character “A”.

TRIGger:A:SERIAL:LOCKLen

This command sets or queries the length in bits of the repeating bit pattern for pattern lock trigger.

Group Trigger

Syntax
```
TRIGger:A:SERIAL:LOCKLen <NR1>
TRIGger:A:SERIAL:LOCKLen?
```

Arguments <NR1> specifies the length in bits. The valid values are 1 to 2147483647.

Examples TRIGGER:A:SERIAL:LOCKLEN 2640 sets the repeating bit pattern length to 2640 bits.

TRIGGER:A:SERIAL:LOCKLEN? might return :TRIGGER:A:SERIAL:LOCKLEN 10000, indicating that the pattern string is 10000 bits.

TRIGger:A:SERIAL:LOCKOffset

This command sets or queries the current bit offset into the pattern lock trigger bit pattern.

Group Trigger

Syntax
```
TRIGger:A:SERIAL:LOCKOffset <NR1>
TRIGger:A:SERIAL:LOCKOffset?
```

Arguments <NR1> specifies the bit offset. The valid values are 1 to 2147483646.

Examples TRIGGER:A:SERIAL:LOCKOFFSET 10 sets the current bit offset to 10 bits.

TRIGGER:A:SERIAL:LOCKOFFSET? might return :TRIGGER:A:SERIAL:LOCKOFFSET 20, indicating that the current bit offset is 20 bits.

TRIGger:A:SERIAL:SOUrce

This command sets or queries the serial data source channel.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:SOUrce {CH1|CH2|CH3|CH4}
TRIGger:A:SERIAL:SOUrce?
```

Arguments CH1–CH4 specifies one of the input channels.

Examples TRIGGER:A:SERIAL:SOURCE CH1 sets the serial data source channel to CH1.

TRIGGER:A:SERIAL:SOURCE? might return :TRIGGER:A:SERIAL:SOURCE CH4, indicating that the serial data source channel is CH4.

TRIGger:A:SERIAL:STANdard

This command sets or queries the standard that identifies the code and bit rate.

Group Trigger

Syntax

```
TRIGger:A:SERIAL:STANdard
{FC133|FC266|FC531|FC1063|D1|D2|VIDEO270|VIDEO360|OC1|OC3|
OC12|ENET1250|FW1394BS400B|FW1394BS800B|CUSTom|ENET100FX|
RIO_500M|RIO_750M|RIO_1G|RIO_SERIAL_1G|VSROC192|ENETXAUI|
SAS3_?|PCIExpress|INFINIBAND|RIO_SERIAL_2G|RIO_SERIAL_3G|
FC2125|RIO_2G|FW1394BS1600B|SAS1_5|ENETXAUI|ENETXAUI2|
FC2125|FC4250|FW1394BS1600B|INFINIBAND|PCIExpress|PCIExpress2|
RIO_2G|RIO_500M|RIO_750M|RIO_SERIAL_1G|RIO_SERIAL_2_5G|
RIO_SERIAL_3G|SAS6_0|SATA1_5|SATA3_0|SATA6_0}
TRIGger:A:SERIAL:STANdard?
```

Related Commands

Arguments FC133: NRZ, 132.8 Mb/s

FC266: NRZ, 265.6 Mb/s

FC531: NRZ, 531.2 Mb/s

FC1063: NRZ, 1.063 Gb/s

OC1: NRZ, 51.84 Mb/s

OC3: NRZ, 155.5 Mb/s

OC12: NRZ, 622.1 Mb/s

ENET1250: NRZ, 3.125 Gb/s

ENETXAUI: NRZ, 1.25 Gb/s

ENETXAUI2: NRZ, 6.25 Gb/s

FW1394BS400B: NRZ, 491.5 Mb/s

FW1394BS1600B: NRZ, 1.966 Gb/s

FW1394BS800B: NRZ, 983.0 Mb/s

CUSTom: NRZ, 155.5 Mb/s

ENET100FX: MLT3, 100 Mb/s

RIO_500M: NRZ, 500 Mb/s, triggers RIO_DRV500, RIO_EDRV500M, and RIO_RCV500, 500Mb/s

RIO_750M: NRZ, 750 Mb/s, triggers RIO_DRV750, RIO_EDRV750M, and RIO_RCV750, 750Mb/s

RIO_1G: NRZ, 1.0 Gb/s, triggers RIO_1G

RIO_SERIAL_1G: NRZ, 1.25 Gb/s, triggers RIO_Serial_1G

RIO_2G: NRZ, 2.0 Gb/s

RIO_SERIAL_2_5G: NRZ, 2.5 Gb/s, triggers RIO_Serial_1G

RIO_SERIAL_3G: NRZ, 3.0 Gb/s, triggers RIO_Serial_1G

VSROC192: NRZ, 1.2441 Gb/s, triggers VSR OC192/STM64

All of the standards might not be available on every instrument.

FC2125: NRZ, 2.125 Gb/s

FC4250: NRZ, 4.25 Gb/s

INFINIBAND: NRZ, 2.5 Gb/s

PCIExpress: NRZ, 2.5 Gb/s

PCIExpress2: NRZ, 5.0 Gb/s

SAS6_0: NRZ, SAS6_0

SATA1_5: NRZ, 1.5 Gb/s

SATA3_0: NRZ, 3.0 Gb/s

SATA6_0: NRZ, 6.0 Gb/s

Examples `TRIGGER:A:SERIAL:STANDARD OC1` sets the standard that identifies the code and bit rate to NRZ, 51.84 Mb/s.

`TRIGGER:A:SERIAL:STANDARD?` might return `:TRIGGER:A:SERIAL:STANDARD CUSTOM`, indicating that the standard that identifies the code and bit rate is set to NRZ, 155.5 Mb/s.

TRIGger:A:SERIAL:TRIGgeron

This command sets or queries the trigger on a designated arbitrary bit pattern or lock on a repeating pattern of known length.

Group Trigger

Syntax
```
TRIGger:A:SERIAL:TRIGgeron {PATtern|LOck}
TRIGger:A:SERIAL:TRIGgeron?
```

Arguments `PATtern` specifies the trigger on a designated arbitrary bit pattern.

`LOck` specifies the trigger locks a repeating pattern of known length.

Examples `TRIGGER:A:SERIAL:TRIGGERON PATTERN` sets trigger on a designated arbitrary bit pattern.

`TRIGGER:A:SERIAL:TRIGGERON?` might return `:TRIGGER:A:SERIAL:TRIGGERON LOCK`, indicating that the trigger locks a repeating pattern of known length.

TRIGger:A:SPI:CONDition

This command sets or queries the trigger condition for SPI triggering.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:CONDition {MISO|MOSI|MISOMOSI}
TRIGger:A:SPI:CONDition?
```

Arguments `MISO` specifies the MISO condition for SPI triggering.

`MOSI` specifies the MISI condition for SPI triggering.

MISOMOSI specifies either the MISO or MISI condition for SPI triggering.

Examples

TRIGGER:A:SPI:CONDITION MOSI sets the SPI trigger condition to MOSI.

TRIGGER:A:SPI:CONDITION? might return :TRIGGER:A:SPI:CONDITION MISI, indicating that the SPI trigger condition is set to MISI.

TRIGger:A:SPI:DATa:MISO:ACTIVE

This command sets or queries the SPI MISO polarity.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:SPI:DATa:MISO:ACTIVE {HIGH|LOW}
TRIGger:A:SPI:DATa:MISO:ACTIVE?
```

Arguments

HIGH specifies HIGH polarity.

LOW specifies LOW polarity.

Examples

TRIGGER:A:SPI:DATA:MISO:ACTIVE HIGH sets the SPI trigger MISO polarity to high.

TRIGGER:A:SPI:DATA:MISO:ACTIVE? might return :TRIGGER:A:SPI:DATA:MISO:ACTIVE LOW, indicating that the SPI MISO polarity is set to low.

TRIGger:A:SPI:DATa:MISO:LEVel

This command sets or queries the threshold for the SPI MISO data source.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:SPI:DATa:MISO:LEVel <NR3>
TRIGger:A:SPI:DATa:MISO:LEVel?
```

Arguments <NR3> specifies the threshold for the SPI trigger MISO data source. The range is ±5.0.

Examples TRIGGER:A:SPI:DATA:MISO:LEVEL 0.0 sets the threshold for the SPI trigger MISO data source to 0.0.

TRIGGER:A:SPI:DATA:MISO:LEVEL? might return :TRIGGER:A:SPI:DATA:MISO:LEVEL 5.0, indicating that the threshold for the SPI MISO data source is set to 5.0.

TRIGger:A:SPI:DATa:MISO:SOUrce

This command sets or queries the MISO data source for the SPI trigger.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:DATa:MISO:SOUrce CH<x>
TRIGger:A:SPI:DATa:MISO:SOUrce?
```

Arguments CH<x> specifies the source for the SPI trigger MISO data source. x can be 1, 2, 3, or 4.

Examples TRIGGER:A:SPI:DATA:MISO:SOURCE CH2 sets the SPI trigger MISO data source to channel CH2.

TRIGGER:A:SPI:DATA:MISO:SOURCE? might return :TRIGGER:A:SPI:DATA:MISO:SOURCE CH1, indicating that the SPI trigger MISO data source is set to CH1.

TRIGger:A:SPI:DATa:MISO:VALue

This command sets or queries the binary data string used for the SPI trigger if the trigger condition is set to MISO or MISOMOSI.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax

```
TRIGger:A:SPI:DATa:MISO:VALue <QString>
TRIGger:A:SPI:DATa:MISO:VALue?
```

Arguments

<QString> specifies the binary data string for the SPI data. You can specify up to 32 bits.

Examples

TRIGGER:A:SPI:DATA:MISO:VALUE "1011" sets the binary data string used for the SPI trigger to 1011.

TRIGGER:A:SPI:DATA:MISO:VALUE? might return :TRIGGER:A:SPI:DATA:MISO:VALUE "10111", indicating that the binary data string used for the SPI trigger is set to 10111.

TRIGger:A:SPI:DATa:MOSI:ACTIVE

This command sets or queries the SPI MOSI polarity.

NOTE. *Not available on some models.*

Group

Low Speed Serial Trigger

Syntax

```
TRIGger:A:SPI:DATa:MOSI:ACTIVE {HIGH|LOW}
TRIGger:A:SPI:DATa:MOSI:ACTIVE?
```

Arguments

HIGH specifies HIGH polarity.

LOW specifies LOW polarity.

Examples

TRIGGER:A:SPI:DATA:MOSI:ACTIVE HIGH sets the SPI trigger MOSI polarity to HIGH.

TRIGGER:A:SPI:DATA:MOSI:ACTIVE? might return :TRIGGER:A:SPI:DATA:MISO:ACTIVE LOW, indicating that the SPI MOSI polarity is set to LOW.

TRIGger:A:SPI:DATa:MOSI:LEVel

This command sets or queries the threshold for the SPI MOSI data source.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:DATa:MOSI:LEVel <NR3>
TRIGger:A:SPI:DATa:MOSI:LEVel?
```

Arguments <NR3> specifies the threshold for the SPI trigger MOSI data source. The range is ±5.0.

Examples TRIGGER:A:SPI:DATA:MOSI:LEVEL 0.0 sets the threshold for the SPI trigger MOSI data source to 0.0.

TRIGGER:A:SPI:DATA:MOSI:LEVEL? might return :TRIGGER:A:SPI:DATA:MISO:LEVEL 5.0, indicating that the threshold for the SPI MOSI data source is set to 5.0.

TRIGger:A:SPI:DATa:MOSI:SOUrce

This command sets or queries the MOSI data source for the SPI trigger.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:DATa:MOSI:SOUrce CH<x>
TRIGger:A:SPI:DATa:MOSI:SOUrce?
```

Arguments CH<x> specifies the source for the SPI trigger MOSI data source. x can be 1, 2, 3, or 4.

Examples TRIGGER:A:SPI:DATA:MOSI:SOURCE CH2 sets the SPI trigger MOSI data source to channel CH2.

TRIGGER:A:SPI:DATA:MOSI:SOURCE? might return :TRIGGER:A:SPI:DATA:MOSI:SOURCE CH1, indicating that the SPI trigger MOSI data source is set to channel CH1.

TRIGger:A:SPI:DATa:MOSI:VALue

This command sets or queries the binary data string used for the SPI trigger if the trigger condition is set to MOSI or MISOMOSI.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:DATa:MOSI:VALue <QString>
TRIGger:A:SPI:DATa:MOSI:VALue?
```

Arguments <QString> specifies the binary data string for the SPI data. You can specify up to 32 bits.

Examples TRIGGER:A:SPI:DATA:MOSI:VALUE "1011" sets the binary data string used for the SPI trigger to 1011.

TRIGGER:A:SPI:DATA:MOSI:VALUE? might return :TRIGGER:A:SPI:DATA:MOSI:VALUE "10111", indicating that the binary data string used for the SPI trigger is set to 10111.

TRIGger:A:SPI:DATa:STARt

This command sets or queries the nth data byte on the data source after the signal on the enable slave source switches to the polarity specified by the Slave Select Polarity. The instrument triggers if the pattern Matches beginning at this point.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:DATa:STARt <NR3>
TRIGger:A:SPI:DATa:STARt?
```

Arguments <NR3> specifies the SPI start byte data. The data can range from 0 to 2^{16}–1.

Examples TRIGGER:A:SPI:DATA:START 0 sets the SPI trigger data start to 0.

TRIGGER:A:SPI:DATA:START? might return :TRIGGER:A:SPI:DATA:START 1, indicating that the SPI trigger data is set to 1.

TRIGger:A:SPI:FORMat

This command sets or queries the SPI trigger data format.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:FORMat {BINary|HEX}
TRIGger:A:SPI:FORMat?
```

Arguments `BINary` specifies a binary data format.

`HEX` specifies a hexadecimal data format.

Examples `TRIGGER:A:SPI:FORMAT BINARY` sets the SPI trigger data format to binary.

`TRIGGER:A:SPI:FORMAT?` might return `:TRIGGER:A:SPI:FORMAT 1`, indicating that the SPI trigger data format is set to HEX.

TRIGger:A:SPI:SCLK:ACTIVE

This command sets or queries the SPI SCLK polarity.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:SCLK:ACTIVE {RISe|FALL}
TRIGger:A:SPI:SCLK:ACTIVE?
```

Arguments `RISe` specifies rising clock polarity.

`FALL` specifies falling clock polarity.

Examples `TRIGGER:A:SPI:SCLK:ACTIVE FALL` sets the SPI SCLK polarity to fall.

`TRIGGER:A:SPI:SCLK:ACTIVE?` might return `:TRIGGER:A:SPI:SCLK:ACTIVE FALL`, indicating that the SPI SCLK polarity is set to FALL.

TRIGger:A:SPI:SCLK:LEVel

This command sets or queries the threshold for the SPI trigger SCLK.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:SCLK:LEVel <NR3>
TRIGger:A:SPI:SCLK:LEVel?
```

Arguments <NR3> specifies the SPI trigger SCLK threshold.

Examples TRIGGER:A:SPI:SCLK:LEVEL 0.0 sets the SPI SCLK threshold to 0.0.

TRIGGER:A:SPI:SCLK:LEVEL? might return :TRIGGER:A:SPI:SCLK:LEVEL 5.0, indicating that the SPI SCLK threshold is set to 5.0.

TRIGger:A:SPI:SCLK:SOUrce

This command sets or queries the SPI SCLK source.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:SCLK:SOUrce CH<x>
TRIGger:A:SPI:SCLK:SOUrce?
```

Arguments CH<x> specifies the SPI SCLK source. x can be 1, 2, 3, or 4.

Examples TRIGGER:A:SPI:SCLK:SOURCE CH2 sets the SPI SCLK trigger source to channel CH2.

TRIGGER:A:SPI:SCLK:SOURCE? might return :TRIGGER:A:SPI:SCLK:SOURCE CH4, indicating that the SPI SCLK trigger source is set to channel CH4.

TRIGger:A:SPI:SS:ACTIVE

This command sets or queries the SPI trigger Slave Select (SS) polarity.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:SS:ACTIVE {HIGH|LOW}
TRIGger:A:SPI:SS:ACTIVE?
```

Arguments `HIGH` specifies HIGH for SS polarity.

`LOW` specifies LOW for SS polarity.

Examples `TRIGGER:A:SPI:SS:ACTIVE HIGH` sets the SPI trigger SS polarity to HIGH.

`TRIGGER:A:SPI:SS:ACTIVE?` might return `:TRIGGER:A:SPI:SS:ACTIVE LOW`, indicating that the SPI trigger SS polarity is set to LOW.

TRIGger:A:SPI:SS:LEVel

This command sets or queries the threshold for the SPI trigger Slave Select (SS) signal.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:SS:LEVel <NR3>
TRIGger:A:SPI:SS:LEVel?
```

Arguments `<NR3>` specifies the threshold for the SPI trigger SS line. The threshold range is ±5.0.

Examples `TRIGGER:A:SPI:SS:LEVEL0.0` sets the threshold level for the SPI trigger SS to 0.0.

`TRIGGER:A:SPI:SS:LEVEL?` might return `:TRIGGER:A:SPI:SS:LEVEL 5.0000`, indicating that the threshold level for the SPI trigger SS is set to 5.0.

TRIGger:A:SPI:SS:SOUrce

This command sets or queries the SPI trigger Slave Select (SS) source.

NOTE. *Not available on some models.*

Group Low Speed Serial Trigger

Syntax
```
TRIGger:A:SPI:SS:SOUrce CH<x>
TRIGger:A:SPI:SS:SOUrce?
```

Arguments CH<x> specifies the source for the SPI SS signal. x can be 1, 2, 3, or 4.

Examples TRIGGER:A:SPI:SS:SOURCE CH2 sets the SPI trigger Slave Select (SS) source to channel CH2.

TRIGGER:A:SPI:SS:SOURCE? might return :TRIGGER:A:SPI:SS:SOURCE CH1, indicating that the SPI trigger Slave Select (SS) source is set to channel CH1.

TRIGger:A:VIDeo? (Query Only)

This query-only command returns the A trigger video parameters.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:VIDeo?
```

Examples TRIGGER:A:VIDEO? might return :TRIGGER:A:VIDEO:CUSTOM:FORMAT INTERLACED; SCAN RATE1; :TRIGGER:A:VIDEO:FIELD ALLFIELDS; HOLDOFF:FIELD 1.0000;TIME 20.0000E-3; :TRIGGER:A:VIDEO:LINE 1;SCAN RATE1; SOURCE CH1;STANDARD NTSC.

TRIGger:A:VIDeo:CUSTom? (Query Only)

This query-only command returns the A trigger custom video parameters.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:VIDeo:CUSTom?
```

Examples `TRIGGER:A:VIDEO:CUSTOM?` might return the parameters `:TRIGGER:A:VIDEO:CUSTOM:FORMAT INTERLACED; SCAN RATE1`.

TRIGger:A:VIDeo:CUSTom:FORMat

This command sets or queries the video trigger format. Use this command only when the video format is set to custom.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:VIDeo:CUSTom:FORMat {INTERLACed|PROGressive}
TRIGger:A:VIDeo:CUSTom:FORMat?
```

Arguments `INTERLACed` argument sets the format for interlaced video lines.

`PROGressive` argument sets the format for progressive video lines.

Examples `TRIGGER:A:VIDEO:CUSTOM:FORMATPROGRESSIVE` sets the custom format for the A video trigger to progressive lines.

`TRIGGER:A:VIDEO:CUSTOM:FORMAT?` might return `:TRIGGER:A:VIDEO:CUSTOM:FORMAT INTERLACED`, indicating that interlaced is selected as the custom format for the A video trigger.

TRIGger:A:VIDeo:CUSTom:LINEPeriod

This command sets or queries the period horizontal line scan. Use this command only when the video format is set to Bilevel or Trilevel. This is equivalent to selecting Video Setup from the Trig menu, setting Bilevel or Trilevel for Format, and setting the Line Period.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:VIDeo:CUSTom:LINEPeriod <NR3>
TRIGger:A:VIDeo:CUSTom:LINEPeriod?
```

Arguments <NR3> is the time for each horizontal scan line.

Examples TRIGGER:A:VIDEO:CUSTOM:LINEPERIOD 63.6E-6 sets the custom line period to 63.6 microseconds.

TRIGGER:A:VIDEO:CUSTOM:LINEPERIOD? might return TRIGGER:A:VIDEO:CUSTOM:LINEPERIOD63.6000E-6, indicating that the line period is set to 63 microseconds.

TRIGger:A:VIDeo:CUSTom:SYNCInterval

This command sets or queries the horizontal line scan rate of the A video trigger. Use this command only when the video format is set to Bilevel. This is equivalent to selecting Video Setup from the Trig menu, setting Bilevel for Format, and setting the sync interval.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:VIDeo:CUSTom:SYNCInterval <NR3>
TRIGger:A:VIDeo:CUSTom:SYNCInterval?
```

Arguments <NR3> is the sync interval for the Bilevel format.

Examples TRIGGER:A:VIDEO:CUSTOM:SYNCINTERVAL 4.7E-6 sets the sync interval to 4.7 microseconds.

TRIGGER:A:VIDEO:CUSTOM:SYNCINTERVAL? might return TRIGGER:A:VIDEO:CUSTOM:SYNCINTERVAL 4.7000E-6, indicating that the custom sync interval is set to 4.7 microseconds.

TRIGger:A:VIDeo:FIELD

This command sets or queries the video field or line that the trigger detects.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:VIDeo:FIELD
{ODD|EVEN|FIELD1|FIELD2|ALLFields|ALLLines|NUMERic}
TRIGger:A:VIDeo:FIELD?
```

Arguments

ODD argument sets the instrument to trigger on interlaced video odd fields.

EVEN argument sets the instrument to trigger on interlaced video even fields.

FIELD1 argument sets the instrument to trigger on interlaced video odd fields (same as ODD).

FIELD2 argument sets the instrument to trigger on interlaced video even fields (same as EVEN).

ALLFields argument sets the instrument to trigger on all fields.

ALLLines argument sets the instrument to trigger on all video lines.

NUMERic argument sets the instrument to trigger on the video signal line specified by the TRIGger:A:VIDeo:LINE command.

Examples

TRIGGER:A:VIDEO:FIELDEVEN 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:A:VIDeo:HOLdoff:FIELD

This command sets or queries the video trigger holdoff in terms of video fields.

NOTE. *Not available on some models.*

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 instrument waits before rearming the video trigger.

Examples

TRIGGER:A:VIDEO:HOLDOFF:FIELD 4.5 sets the instrument to wait 4.5 video fields before rearming the trigger.

TRIGGER:A:VIDEO:HOLDOFF:FIELD? might return :TRIGGER:A:VIDEO:HOLDOFF:FIELD 5, indicating that the instrument is set to wait 5 video fields before rearming the trigger.

TRIGger:A:VIDeo:LINE

This command sets or queries the video line number on which the instrument triggers. This command is equivalent to selecting Video Setup from the Trig menu, selecting Line # in the Trigger on box, and setting the line number. Use the TRIGger:A:VIDeo:FIELD command to actually trigger the instrument on the line that you specify with this command.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:VIDeo:LINE <NR1>
TRIGger:A:VIDeo:LINE?
```

Related Commands TRIGger:A:VIDeo:FIELD

Arguments <NR1> argument is an integer that sets the video line number on which the instrument triggers. The following table lists the valid choices, depending on the active video standard.

Table 2-50: Video Line Numbering Ranges

Video Standard	Line Number Range
CUSTOM	4–3000
NTSC	1–263 (odd) and 264–525 (even)
PAL	1–625
SECAM	1–625
HD480P60	1–520
HD720P60	1–750
HD1080150	1–1125
HD1080160	1–1125
HD1080P24	1–1125
HD1080P25	1–1125
HD1080SF24	1–1125

Examples `TRIGGER:A:VIDEO:LINE 23` sets the instrument to trigger on the line 23.

`TRIGGER:A:VIDEO:LINE?` might return `:TRIGGER:A:VIDEO:LINE 10`, indicating that the instrument is set to trigger on line 10.

TRIGger:A:VIDeo:POLarity

This command sets or queries the polarity of the A video trigger.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:VIDeo:POLarity {INVERTed|NORMal}
TRIGger:A:VIDeo:POLarity?
```

Arguments `INVERTed` argument sets the instrument to trigger on a positive video sync pulse.

`NORMal` argument sets the instrument to trigger on a negative video sync pulse.

Examples `TRIGGER:A:VIDEO:POLARITY NORMAL` sets the instrument to trigger on a negative video pulse.

`TRIGGER:A:VIDEO:POLARITY?` might return `:TRIGGER:A:VIDEO:POLARITY INVERTED`, indicating that the instrument is set to trigger on a positive video sync pulse.

TRIGger:A:VIDeo:SCAN

This command sets or queries the video trigger horizontal line scan rate. This command is for compatibility with earlier instruments.

NOTE. *Not available on some models.*

Group Trigger

Syntax
```
TRIGger:A:VIDeo:SCAN {RATE1|RATE2|RATE3|RATE4|RATE5}
TRIGger:A:VIDeo:SCAN?
```

Arguments

RATE1 argument sets the range of the video line scan rate to 15 kHz through 20 kHz. This is the standard broadcast rate.

RATE2 argument sets the range of the video line scan rate to 20 kHz through 25 kHz.

RATE3 argument sets the range of the video line scan rate to 25 kHz through 35 kHz.

RATE4 argument sets the range of the video line scan rate to 35 kHz through 50 kHz.

RATE5 argument sets the range of the video line scan rate to 50 kHz through 65 kHz.

Examples

TRIGGER:A:VIDEO:SCAN RATE1 sets the scan rate of the A video trigger to Rate 1, which is 15 kHz to 20 kHz (standard broadcast rate).

TRIGGER:A:VIDEO:SCAN? might return :TRIGGER:A:VIDEO:SCAN RATE2, indicating that the video line rate for the A trigger is set to Rate 2, which is 20 kHz to 23 kHz.

TRIGger:A:VIDeo:SOUrce

This command sets or queries the source for the A video trigger. This command is equivalent to selecting Video Setup from the Trig menu and selecting a channel from the Source drop-down menu.

NOTE. *Not available on some models.*

Group

Trigger

Syntax

```
TRIGger:A:VIDeo:SOUrce CH<x>
TRIGger:A:VIDeo:SOUrce?
```

Arguments

CH<x> argument specifies one of the input channels of the instrument as the A video trigger. The value of x ranges from 1 through 4.

Examples

TRIGGER:A:VIDEO:SOURCECH1 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

This command sets or queries the video standard.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:A:VIDeo:STANdard
{BILevelcustom|TRILevelcustom|NTSc|PAL|
SECAM|HD480P60|HD576P50|HD875I60|HD720P30|HD720P50|
HD720P60|HD1080I50|HD1080I60|HD1080P24|
HD1080P25|HD1080P30|HD1080P50|HD1080P60| HD1080SF24}
TRIGger:A:VIDeo:STANdard?
```

Arguments

`BILevelcustom` argument sets the instrument to use custom video parameters that you set with the TRIGger:A:VIDeo:CUSTom:SYNCInterval command.

`TRILevelcustom` argument sets the instrument to use custom video horizontal scan rate parameters that you set with the TRIGger:A:VIDeo:CUSTom:LINEPeriod command.

`NTSc` argument sets the instrument 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` argument sets the instrument 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` argument sets the instrument to trigger on video signals that meet the SECAM standard.

`HD480P60` argument sets the instrument to trigger on the HDTV 480/60 progressive format.

`HD576P50` argument sets the instrument to trigger on the HDTV 576/50 progressive format.

`HD875I60` argument sets the instrument to trigger on the HDTV 875/60 format.

`HD720P30` argument sets the instrument to trigger on the HDTV 720/30 progressive format.

`HD720P50` argument sets the instrument to trigger on the HDTV 720/50 progressive format.

`HD720P60` argument sets the instrument to trigger on the HDTV 720/60 progressive format.

`HD1080I50` argument sets the instrument to trigger on HDTV 1080/50 interlaced format.

HD1080I60 argument sets the instrument to trigger on HDTV 1080/60 interlaced format.

HD1080P24 argument sets the instrument to trigger on HDTV 1080/24 progressive format.

HD1080P25 argument sets the instrument to trigger on HDTV 1080/25 progressive format.

HD1080P30 argument sets the instrument to trigger on HDTV 1080/30 progressive format.

HD1080P50 argument sets the instrument to trigger on HDTV 1080/50 progressive format.

HD1080P60 argument sets the instrument to trigger on HDTV 1080/60 progressive format.

HD1080SF24 argument sets the instrument to trigger on HDTV 1080/24 segmented frame format.

Examples

TRIGGER:A:VIDEO:STANDARD NTSC sets the instrument to trigger on NTSC-standard video signals.

TRIGGER:A:VIDEO:STANDARD? might return :TRIGGER:A:VIDEO:STANDARD NTSC, indicating that the standard for the A video trigger is set to NTSC.

TRIGger:AUXLevel

This command sets or queries the auxiliary (Aux) level for the edge trigger. This command supersedes any other argument that sets the auxiliary trigger level.

Group

Trigger

Syntax

```
TRIGger:AUXLevel {<NR3>|ECL|TTL}
TRIGger:AUXLevel?
```

Related Commands

TRIGger:{A|B}:EDGE:SLOpe, TRIGger:{A|B}:EDGE:SOUrce

Arguments

<NR3> is the auxiliary level in volts.

ECL specifies a preset ECL level.

TTL specifies a preset TTL level.

Examples `TRIGGER:AUXLEVEL TTL` sets the auxiliary level for the edge trigger to TTL.

`TRIGGER:AUXLEVEL?` might return `:TRIGGER:AUXLEVEL TTL`, indicating that the auxiliary level for the edge trigger is set to TTL.

TRIGger:B (No Query Form)

This command sets the B trigger level to 50% of minimum and maximum. The query form of this command returns the B trigger parameters. This command is similar to selecting B Event (Delayed) Trigger Setup from the Trig menu and then viewing the current setups.

Group Trigger

Syntax `TRIGger:B SETLevel`

Related Commands TRIGger:A TRIGger:B:STATE

Arguments `SETLevel` sets the B trigger level to 50% of MIN and MAX.

Examples `TRIGGER:B SETLEVEL` sets the B trigger level to 50% of MIN and MAX.

`TRIGGER:B?` might return the following B trigger parameters:
`:TRIGGER:B:STATE 0;TYPE EDGE; LEVEL -220.0000E-3;BY TIME;EDGE:SOURCE CH1;SLOPE RISE;COUPLING DC; :TRIGGER:B:TIME 16.0000E-9;EVENTS:COUNT 2`.

TRIGger:B:BY

This command selects or returns whether the B trigger occurs after a specified number of events or a specified period of time after the A trigger. This is equivalent to selecting B Event (Delayed) Trigger Setup from the Trig menu, selecting the A →B Seq tab, and then choosing Trig After Time or Trig on nth event.

NOTE. *The traditional Runs After functionality is now served by the Horizontal Delay function. For details, see the HORizontal[:MAIn]:DELay:MODe and HORizontal[:MAIn]:DELay:TIMe commands.*

Group Trigger

Syntax

```
TRIGger:B:BY {EVENTS|TIMe}
TRIGger:B:BY?
```

Related Commands

TRIGger:B:EVENTS:COUNt, TRIGger:B:TIMe, HORizontal[:MAIn]:DELay:MODe, HORizontal[:MAIn]:DELay:TIMe TRIGger:B:STATE

Arguments

`EVENTS` sets the B trigger to take place following a set number of trigger events after the A trigger occurs. The number of events is specified by `TRIGger:B:EVENTS:COUNt`.

`TIMe` sets the B trigger to occur a set time after the A trigger event. The time period is specified by `TRIGger:B:TIMe`.

Examples

`TRIGGER:B:BY TIME` sets the B trigger to occur at a set time after the A trigger event.

`TRIGGER:B:BY?` might return `:TRIGGER:B:BY EVENTS`, indicating that the B trigger takes place following a set number of trigger events after the A trigger occurs.

TRIGger:B:EDGE:ENVelope

This command sets or queries whether the B edge trigger will use envelope triggering. DPO70000SX Series only.

Group

Trigger

Syntax

```
TRIGger:B:EDGE:ENVelope <OFF|ON>
TRIGger:B:EDGE:ENVelope?
```

Arguments

`OFF` disables edge envelope triggering.

`ON` enables edge envelope triggering.

Examples

`TRIGGER:B:EDGE:ENVELOPE ON` enables edge envelope triggering.

`TRIGGER:B:EDGE:ENVELOPE?` may return ON or OFF.

TRIGger:B:EVENTS? (Query Only)

This query-only command returns the current B trigger events parameter. This command is equivalent to selecting B Event (Delayed) Trigger Setup from the

Trig menu, selecting the A →B Seq tab, choosing Trig on nth event, and viewing the Trig Event setting.

Group Trigger

Syntax `TRIGger:B:EVENTS?`

Related Commands TRIGger:B:EVENTS:COUNt, TRIGger:B:STATE

Examples `TRIGGER:B:EVENTS?` might return

`:TRIGGER:B:EVENTS:COUNT 2`, indicating that the two events must occur before the B trigger occurs.

TRIGger:B:EVENTS:COUNt

This command sets or queries the number of events that must occur before the B trigger (when TRIG:DELay:BY is set to EVENTS). This command is equivalent to selecting B Event (Delayed) Trigger Setup from the Trig menu, selecting the A →B Seq tab, choosing Trig on nth event, and setting the desired Trig Event value.

Group Trigger

Syntax
```
TRIGger:B:EVENTS:COUNt <NR1>
TRIGger:B:EVENTS:COUNt?
```

Related Commands TRIGger:B:EVENTS?, TRIGger:B:STATE

Arguments `<NR1>` is the number of B trigger events, which can range from 1 to 10,000,000.

Examples `TRIGGER:B:EVENTS:COUNT 4` sets the number of B trigger events to four.

`TRIGGER:B:EVENTS:COUNT?` might return `:TRIGGER:B:EVENTS:COUNT 2`, indicating that two events must occur after the A trigger before the B trigger can occur.

TRIGger:B:PULse:GLItch:ENVelope

This command sets or queries whether the B glitch trigger will use envelope triggering. DPO70000SX Series only.

Group Trigger

Syntax
```
TRIGger:B:PULse:GLItch:ENVelope <OFF|ON>
TRIGger:B:PULse:GLItch:ENVelope?
```

Arguments `OFF` disables glitch envelope triggering.

`ON` enables glitch envelope triggering.

Examples `TRIGGER:B:PULSE:GLITCH:ENVELOPE ON` enables glitch envelope triggering.

`TRIGGER:B:PULSE:GLITCH:ENVELOPE?` may return ON or OFF.

TRIGger:B:PULse:PERiod? (Query Only)

This query-only command returns the Pulse Trigger settings specific to Period/Frequency trigger. These include the array of high and low limits, the trigger when enumeration, the polarity enumeration, the qualification enumeration, and the view enumeration. The view enumeration controls whether the high and low limits are expressed in units of time (Seconds) or frequency (Hz). The qualification enumeration determines if the trigger is state-qualified by additional signals (channels). The polarity enumeration determines whether the rising or falling edges of the signal are used to initiate the trigger. The trigger when enumeration determines if the trigger should occur when the period (frequency) is greater than or less than a single (low) limit value, or within or outside the period (frequency) of a high-low limit period (frequency) pair.

The Period/Frequency Trigger is similar to Width Trigger, but instead of triggering on a single pulse of a given width, it triggers on the period (two back-to-back pulses, one up and one down) of a given width. As a convenience the period limits (Seconds) can be expressed as frequency limits (Hz) by changing the view enumeration. It is often used to determine if oscillators are operating within specification.

Group Trigger

Syntax
```
TRIGger:B:PULse:PERiod?
```

Related Commands TRIGger:A:PULse:PERiod?TRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:POLarityTRIGger:A:PULse:PERiod:QUALifyTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod?TRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:

WHEnTRIGger:B:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:QUAlifyTRIGger:B:PULse:PERiod:VIEW

Examples `TRIGGER:B:PULSE:PERIOD?` might return `:TRIGGER:B:PULSE:PERIOD:'90.0000E-9;10.0000;GREATERTHAN;POSITIVE;OCCURS\n'`, indicating the period limits, when enumeration, polarity, and so forth.

TRIGger:B:PULse:PERiod:HIGHLimit

This command sets or queries the Period/Frequency high limit. When the VIEW is PERiod the units of this number are time (Seconds). When the VIEW is FREQuency, the units of this number are frequency (Hz). The high limit is used in combination with the low limit to define a range if the WHEn enumeration is WITHin or OUTside. These limits specify the most significant characteristic of the signal for this Pulse Trigger Type.

Group Trigger

Syntax
```
TRIGger:B:PULse:PERiod:HIGHLimit <NR3>
TRIGger:B:PULse:PERiod:HIGHLimit?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:WHEnTRIGger:B:PULse:PERiod:POLarity

Arguments `<NR3>` is the period (in Seconds) or frequency (in Hz).

Examples `TRIGGER:B:PULSE:PERIOD:HIGHLIMIT 101.0e-9` sets the Pulse Period Trigger High Limit to one hundred one nanoseconds, which, in combination with a Low Limit of 99.0e-9, is suitable for triggering on a 10 MHz clock if the WHEn enumeration is set to WITHin.

`TRIGGER:B:PULSE:PERIOD:HIGHLIMIT?` might return `:TRIGGER:B:PULSE:PERIOD:HIGHLIMIT 101.0e-9` (if View is Period), or `:TRIGGER:B:PULSE:PERIOD:HIGHLIMIT 10.1e6` (if View is Frequency).

TRIGger:B:PULse:PERiod:LOWLimit

This command sets or queries the Period/Frequency low limit. When the VIEW is PERiod the units of this number are time (Seconds). When the VIEW is FREQuency, the units of this number are frequency (Hz). The low limit is used alone when the WHEn enumeration is LESSthan or GREATerthan, and is used in combination with the high limit to define a range if the WHEn enumeration is WITHin or OUTside. The limits specify the most significant characteristic of the signal for this Pulse Trigger Type.

Group Trigger

Syntax

```
TRIGger:B:PULse:PERiod:LOWLimit <NR3>
TRIGger:B:PULse:PERiod:LOWLimit?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:WHEnTRIGger:B:PULse:PERiod:POLarity

Arguments <NR3> is the period (in Seconds) or frequency (in Hz).

Examples `TRIGGER:B:PULSE:PERIOD:LOWLIMIT 101.0e-9` sets the Pulse Period Trigger Low Limit to one hundred one nanoseconds, which is suitable for triggering on a 10 MHz clock if the WHEn enumeration is set to LESSthan.

`TRIGGER:B:PULSE:PERIOD:LOWLIMIT?` might return `:TRIGGER:B:PULSE:PERIOD:LOWLIMIT 101.0e-9` (if View is Period), or `:TRIGGER:B:PULSE:PERIOD:LOWLIMIT 9900990`,(if View is Frequency).

TRIGger:B:PULse:PERiod:POLarity

This command sets or queries the polarity for the Pulse Period trigger. It refers to the polarity of the edges that begin and end a given period of the signal, and thus allows duty-cycle testing of clock signals.

Group Trigger

Syntax

```
TRIGger:B:PULse:PERiod:POLarity {NEGAtive|POSITIVe}
TRIGger:B:PULse:PERiod:POLarity?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:POLarityTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:VIEW

Arguments `POSITIVe` specifies to trigger on the rising or positive edge of a signal.

`NEGAtive` specifies to trigger on the falling or negative edge of a signal.

Examples `TRIGGER:B:PULSE:PERIOD:POLARITY POSITIVE` sets the B pulse period trigger slope to positive, which triggers on the rising edge of the signal.

`TRIGGER:B:PULSE:PERIOD:POLARITY?` might return `:TRIGGER:A:PULSEL:PERIOD:POLARITY NEGATIVE`, indicating that the B pulse period trigger polarity is negative.

TRIGger:B:PULse:PERiod:QUAlify

This command sets or queries the Qualification setup for Pulse Period Trigger. The high/low state of one or more other signals (channels) may be used to qualify whether the trigger should occur. The BUS option is not available on the 77K scope family at this time.

When the QUALify enumeration is OCCurs, no qualification is done.

When the QUALify enumeration is LOGIC, the scope triggers using the pulse period trigger definition as qualified by the high/low states of the qualifying signals. Additional description is given elsewhere in this document under the term logic qualification.

Group Trigger

Syntax
```
TRIGger:B:PULse:PERiod:QUAlify {OCCurs | LOGIC | BUS}
TRIGger:B:PULse:PERiod:QUAlify?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:WHEnTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:WHEnTRIGger:B:PULse:PERiod:VIEW

Arguments

`OCCurs` specifies the no other signals are used to qualify the pulse period trigger.

`LOGIC` specifies that other analog signals (channels) are used to qualify the pulse period trigger.

`BUS` specifies that other digital signals (digital channels) are used to qualify the pulse period trigger.

Examples

`TRIGGER:B:PULSE:PERIOD:QUALIFY LOGIC` specifies that one or more other analog signals (channels) are used to qualify the pulse period trigger.

`TRIGGER:B:PULSE:PERIOD:QUALIFY?` might return `:TRIGGER:B:PULSE:PERIOD:QUALIFY OCCURS`, indicating that the B pulse period trigger is not qualified by any other signals (channels).

TRIGger:B:PULse:PERiod:VIEW

This command sets or queries the view for the pulse period trigger. When PERiod is selected the LOWLimit and HIGHLimit values are in units of time (Seconds). When FREQuency is selected, the units are in frequency (Hz). This two methods are equivalent, so the alternative views are provided as a convenience.

Group

Trigger

Syntax

```
TRIGger:B:PULse:PERiod:VIEW {PERiod|FREQuency}
TRIGger:B:PULse:PERiod:VIEW?
```

Related Commands

TRIGger:{A|B}:EDGE:SOUrceTRIGger:{A|B}: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:B:PULSE:PERIOD:VIEW PERIOD` sets the interpretation of the high and low limits to units of time (Seconds).

`TRIGGER:B:PULSE:PERIOD:VIEW?` might return `:TRIGGER:B:PULSE:PERIOD:VIEW FREQUENCY`, indicating that the interpretation of the high and low limits is in units of frequency (Hz).

TRIGger:B:PULse:PERiod:WHEn

This command sets or queries the trigger when enumeration for the Pulse Period Trigger.

If the argument is LESSthan , the scope triggers on the signal when the period (frequency) of the signal is less than the period (frequency) of the Low Limit.

If the argument is GREATerthan, the scope triggers on the signal when the period (frequency) of the signal is greater than the period (frequency) of the Low Limit.

If the argument is WITHin, the scope triggers on the signal when the period (frequency) of the signal is within the range defined by the Low Limit and the High Limit.

If the argument is OUTside, the scope triggers on the signal when the period (frequency) of the signal is outside of the range defined by the Low Limit and the High Limit.

Group Trigger

Syntax

```
TRIGger:B:PULse:PERiod:WHEn {LESSthan | GREATerthan | WITHin
| OUTside}
TRIGger:B:PULse:PERiod:WHEn?
```

Related Commands TRIGger:A:PULse:PERiod:LOWLimitTRIGger:A:PULse:PERiod:HIGHLimitTRIGger:A:PULse:PERiod:POLarityTRIGger:A:PULse:PERiod:VIEWTRIGger:B:PULse:PERiod:LOWLimitTRIGger:B:PULse:PERiod:HIGHLimitTRIGger:B:PULse:PERiod:POLarityTRIGger:B:PULse:PERiod:VIEW

Arguments

`LESSthan` tells the scope to trigger when the period (frequency) of the signal is less than the low limit value.

`GREATerthan` tells the scope to trigger when the period (frequency) of the signal is greater than the low limit value.

`WITHin` tells the scope to trigger when the period (frequency) of the signal is inside the range set by the low and high limit values.

`OUTside` tells the scope to trigger when the period (frequency) of the signal is outside of the range set by the low and high limit values.

Examples `TRIGGER:B:PULSE:PERIOD:WHEN LESSTHAN` tells the scope to trigger when the period (frequency) of the signal is less than the low limit value.

TRIGGER:B:PULSE:PERIOD:WHEN WITHin tells the scope to trigger when the period (frequency) of the signal is inside the range set by the low and high limit values.

TRIGGER:B:PULSE:PERIOD:WHEN? might return LESSTHAN indicating the instrument will trigger when the period (frequency) of the signal is less than the low limit value.

TRIGger:B:PULse:TIMEOut:ENVelope

This command sets or queries whether the B timeout trigger will use envelope triggering. DPO70000SX Series only.

Group Trigger

Syntax
```
TRIGger:B:PULse:TIMEOut:ENVelope <OFF|ON>
TRIGger:B:PULse:TIMEOut:ENVelope?
```

Arguments OFF disables timeout envelope triggering.

ON enables timeout envelope triggering.

Examples TRIGGER:B:PULSE:TIMEOUT:ENVELOPE ON enables timeout envelope triggering.

TRIGGER:B:PULSE:TIMEOUT:ENVELOPE? may return ON or OFF.

TRIGger:B:PULse:WIDth:ENVelope

This command sets or queries whether the B width trigger will use envelope triggering. DPO70000SX Series only.

Group Trigger

Syntax
```
TRIGger:B:PULse:WIDth:ENVelope <OFF|ON>
TRIGger:B:PULse:WIDth:ENVelope?
```

Arguments OFF disables width envelope triggering.

ON enables width envelope triggering.

Examples `TRIGGER:B:PULSE:WIDTH:ENVELOPE OFF` disables width envelope triggering.

`TRIGGER:B:PULSE:WIDTH:ENVELOPE?` may return ON or OFF.

TRIGger:B:RESET:ACCEPTCOUNT? (Query Only)

This command queries the Reset Accept Count for a sequence trigger reset by the ACCept and REJect Reset Types. These are often referred to as the Validation/Invalidation Triggers. The AcceptCount is equivalent to the number of acquired waveforms, and is part of the equation TotalCount = AcceptCount + RejectCount used to form the ratio of accepted to total triggers that is optionally displayed when Validation/Invalidation Triggers are employed. The Accept Count is always equal the Acquisition Count.

For example, if the Reset Type Accept is active, the AcceptCount is the number of triggers that have been accepted, a query only value determined by the trigger system. Any acquired waveform obtained by Reset Type Accept or Reject is positioned at the A-Event.

For example, if Reset Type Reject is active, the AcceptCount is still the number of triggers that have been accepted, a query only value determined by the trigger system. Any acquired waveform obtained by Reset Type Accept or Reject is positioned at the A-Event.

Group Trigger

Syntax `TRIGger:B:RESET:ACCEPTCOUNT?`

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:RATIO?, TRIGger:B:RESET:RATIOENable, TRIGger:B:RESET:REJECTTIMEout, TRIGger:B:RESET:REJECTCOUNT?, TRIGger:B:RESET:TOTALCOUNT?

Arguments None

Examples `TRIGGER:B:RESET:ACCEPTCOUNT?` might return zero or any positive integer that represents the number of triggers accepted when Reset by Accept or Reset by Reject are active. The Accept Count is always equal the Acquisition Count.

TRIGger:B:RESET:ACCEPTTIMEout

This command sets or queries the Reset Accept Time Out timer for a sequence trigger reset by the ACCept Reset Type. This is often referred to as the Validation Trigger. The Accept Time Out indicates the time, as measured from the A-Event

to the B-Event, in which the B-Event must occur in order to accept the A-Event. When the B-Event occurs within the Accept Time Out, the scope triggers on the A-Event, because the B-Event is interpreted as a validation of the A-Event. Note that the acquired waveform trigger position is location at the A-Event, contrary to the behavior of most A->B Sequence triggers. In this way, the B-Event validates the A-Event. If the B-Event does not occur within the Accept Time Out, the scope again starts waiting for an A-Event to occur, i.e., no trigger occurs and the acquisition is not retained.

For example, if the Accept Time Out is set to 5.2 µs, the A-Event will trigger the scope only if the B-Event occurs before 5.2 µs has elapsed after the A-Event. Any acquired waveform is positioned at the A-Event.

Group Trigger

Syntax

```
TRIGger:B:RESET:ACCEPTTIMEout <NR3>
TRIGger:B:RESET:ACCEPTTIMEout?
```

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:RATIO?, TRIGger:B:RESET:RATIOENable, TRIGger:B:RESET:REJECTTIMEout, TRIGger:B:RESET:ACCEPTCOUNT?, TRIGger:B:RESET:REJECTCOUNT?, TRIGger:B:RESET:TOTALCOUNT?

Arguments <NR3> specifies the Accept Time Out time.

Examples `TRIGGER:B:RESET:ACCEPTTIMEout 5.2E-6` sets the trigger Accept Time Out to 5.2e-6 seconds.

`TRIGGER:B:RESET:ACCEPTTIMEout?` might return `:TRIGGER:B:RESET:ACCEPTTIMEout 5.2E-06`, indicating Accept Time Out is 5.2 µs.

TRIGger:B:RESET:RATIO? (Query Only)

This command queries the (floating point) Ratio of accepted to total triggers for a sequence trigger reset by the ACCepts and REJects Reset Types. These are often referred to as the Validation/Invalidation Triggers. The Ratio = Accepted Count / Total Count, and is a query only value generated by the trigger system. The TotalCount = AcceptCount + RejectCount.

For example, if the Reset Type Accept is active, and 100 triggers have been accepted and 100 trigger have been rejected, then the ratio = acceptCount / totalCount, then the ratio is 50%. The ratio is expressed in scientific notation.

Group Trigger

Syntax
```
TRIGger:B:RESET:RATIO?
```

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:REJECTCOUNT?, TRIGger:B:RESET:RATIOENable, TRIGger:B:RESET:REJECTTIMEout, TRIGger:B:RESET:ACCEPTTIMEout, TRIGger:B:RESET:ACCEPTCOUNT?

Arguments None

Examples `TRIGGER:B:RESET:RATIO?` may return zero or any positive floating point value less than 1. If you display the ratio using the user interface, the ratio is displayed as a percent.

TRIGger:B:RESET:RATIOENable

This command sets or queries the Ratio Enable enumerated value (OFF/ON). If ON, a ratio is displayed in the graticule area. The displayed value represents the ratio of accepted to total triggers for a sequence trigger reset by the ACCept and REJect Reset Types. These are often referred to as the Validation/Invalidation Triggers. The default value is ON.

For example, if the Reset Type Accept is active, and RatioEnable is ON, the ratio of accepted to total triggers is displayed in the graticule area as a percent.

Group Trigger

Syntax
```
TRIGger:B:RESET:RATIOENable <ON/OFF>
TRIGger:B:RESET:RATIOENable?
```

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:REJECTCOUNT?, TRIGger:B:RESET:RATIO?, TRIGger:B:RESET:REJECTTIMEout, TRIGger:B:RESET:ACCEPTTIMEout, TRIGger:B:RESET:ACCEPTCOUNT?

Arguments ON enables the display of the ratio of accepted to total triggers in the graticule area.

OFF does not display the ratio of accepted triggers.

Examples `TRIGGER:B:RESET:RATIOENable?` may return OFF or ON.

TRIGger:B:RESET:REJECTCOUNT? (Query Only)

This command queries the Reset Reject Count for a sequence trigger reset by the ACCept and REJect Reset Types. These are often referred to as the Validation/Invalidation Triggers. The RejectCount is the number of triggers rejected when these triggers are active, and is part of the equation TotalCount = AcceptCount + RejectCount used to form the ratio of accepted to total triggers that is optionally displayed when Validation/Invalidation Triggers are employed.

For example, if the Reset Type Accept is active, the RejectCount is the number of triggers that have been rejected, a query only value determined by the trigger system.

For example, if Reset Type Reject is active, the RejectCount is the number of triggers that have been rejected, a query only value determined by the trigger system.

Group Trigger

Syntax `TRIGger:B:RESET:REJECTCOUNT?`

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:RATIO?, TRIGger:B:RESET:RATIOENable, TRIGger:B:RESET:REJECTTIMEout, TRIGger:B:RESET:ACCEPTCOUNT?, TRIGger:B:RESET:TOTALCOUNT?

Arguments None

Examples `TRIGGER:B:RESET:REJECTCOUNT?` query may return zero or any positive integer that represents the number of triggers rejected when the Accept or Reject Reset Types are active. Any acquired waveform obtained by Accept or Reject Reset Types is positioned at the A-Event.

TRIGger:B:RESET:REJECTTIMEout

This command sets or queries the Reset Reject Time Out timer for a sequence trigger reset by the REJect Reset Type. This is often referred to as the Invalidation Trigger. The Reject Time Out indicates the time measured from the A-Event in which the B-Event must occur in order to reject the A-Event. When the B-Event does not occur within the Reject Time Out, the scope triggers on the A-Event, because the B-Event is interpreted as an invalidation or rejection of the A-Event. If the B-Event does occur within the Reject Time Out, the scope again starts waiting for an A-Event to occur, i.e., no trigger occurs and no acquisition is retained. Note that the acquired waveform trigger position is located at the A-Event, contrary to the behavior of most A->B Sequence triggers.

For example, if the Reject Time Out is set to 5.2 µS, the A-Event will trigger the scope only if the B-Event does not occur before 5.2 µS has elapsed after the A-Event. The acquired waveform is positioned at the A-Event. When the B-Event does occur before the 5.2 µs has elapsed, then the scope does not trigger, and again starts waiting for an A-Event to occur. In this way the B-Event invalidates an A-Event.

Group Trigger

Syntax
```
TRIGger:B:RESET:REJECTTIMEout <NR3>
TRIGger:B:RESET:REJECTTIMEout?
```

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:RATIO?, TRIGger:B:RESET:RATIOENable, TRIGger:B:RESET:ACCEPTTIMEout, TRIGger:B:RESET:ACCEPTCOUNT?, TRIGger:B:RESET:REJECTCOUNT?, TRIGger:B:RESET:TOTALCOUNT?

Arguments <NR3> specifies the Reject Time Out time.

Examples TRIGGER:B:RESET:REJECTTIMEout 5.2E-6 sets the trigger reset Reject Time Out to 5.2e-6 seconds.

TRIGGER:B:RESET:REJECTTIMEout? might return :TRIGGER:B:RESET:REJECTTIMEout 5.2E-06, indicating the Reject Time Out is 5.2 µs.

TRIGger:B:RESET:SOUrce

This command sets or queries the trigger source for the A→B sequential trigger reset feature, except for the Timeout trigger type.

Group Trigger

Syntax
```
TRIGger:B:RESET:SOUrce {CH<x>|AUXiliary}
TRIGger:B:RESET:SOUrce?
```

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:THReshold

Arguments CH<x> specifies one of the input channels as the reset source. Input channels are specified by x and can be 1, 2, 3, or 4.

AUXiliary specifies an external trigger (using the Auxiliary Trigger Input connector) as the reset source.

Examples

TRIGGER:B:RESET:SOURCE CH4 sets Channel 4 as the input source for the trigger reset.

TRIGGER:B:RESET:SOURCE? might return :TRIGGER:B:RESET:SOURCE CH1, indicating that the current input source for the trigger reset is Channel 1.

TRIGger:B:RESET:STATE

This command sets or queries the trigger reset state for the A→B sequential State trigger reset feature.

Group

Trigger

Syntax

```
TRIGger:B:RESET:STATE {HIGH|LOW}
TRIGger:B:RESET:STATE?
```

Related Commands

TRIGger:B:RESET:TYPe, TRIGger:B:RESET:SOUrce, TRIGger:B:RESET:THReshold

Arguments

HIGH sets the condition for the state sequential trigger reset to high.

LOW sets the condition for the state sequential trigger reset to low.

Examples

TRIGGER:B:RESET:STATE LOW sets the condition for the state sequential trigger reset to Low.

TRIGGER:B:RESET:STATE? might return :TRIGGER:B:RESET:STATE HIGH, indicating that High is the condition for the state sequential trigger reset.

TRIGger:B:RESET:THReshold

This command sets or queries the trigger threshold for the A→B sequential trigger reset, except for the Timeout reset type.

Group

Trigger

Syntax

```
TRIGger:B:RESET:THReshold {ECL|TTL|<NR3>}
TRIGger:B:RESET:THReshold?
```

Related Commands TRIGger:B:RESET:STATE

Arguments `ECL` sets the reset threshold to ECL voltage levels.

`TTL` sets the reset threshold to TTL voltage levels.

`<NR3>` sets the reset threshold in volts.

Examples `TRIGGER:B:RESET:THRESHOLD TTL` sets the trigger reset threshold level to TTL.

`TRIGGER:B:RESET:THRESHOLD?` might return `:TRIGGER:B:RESET:THRESHOLD TTL`, indicating that the trigger reset threshold level is set to TTL.

TRIGger:B:RESET:TIMEOut

This command sets or queries the reset timer for a sequential timeout trigger reset. For example, if the timeout is set to 1 s, this additional reset time begins following an A trigger event and runs sequentially with any B Trigger Delay. If a B trigger event is not found before it expires, the instrument searches for a new A event trigger sequence.

Group Trigger

Syntax
```
TRIGger:B:RESET:TIMEOut <NR3>
TRIGger:B:RESET:TIMEOut?
```

Related Commands TRIGger:B:RESET:TYPe

Arguments `<NR3>` specifies the reset time.

Examples `TRIGGER:B:RESET:TIMEOUT 1.0E-6` sets the trigger reset timeout to 1 s.

`TRIGGER:B:RESET:TIMEOUT?` might return `:TRIGGER:B:RESET:TIMEOUT 2.0000E-09`, indicating that the trigger reset timeout is 2 ns.

TRIGger:B:RESET:TOTALCOUNT? (Query Only)

This command queries the Reset Total Count for a sequence trigger reset by the ACCept and REJect Reset Types. These are often referred to as the Validation/Invalidation Triggers. The TotalCount is the sum of the accepted and rejected triggers when the Reset or Accept Reset Types are active, and is part of

the equation TotalCount = AcceptCount + RejectCount used to form the ratio of accepted to total triggers that is optionally displayed when Validation/Invalidation Triggers are employed.

For example, if the Reset Type Accept is active, the TotalCount is the sum of the accepted and rejected triggers processed by the Accept Reset Type, a query only value determined by the trigger system. Any acquired waveform obtained by Accept or Reject Reset Types is positioned at the A-Event.

For example, if the Reset Type Reject is active, the TotalCount is the sum of the accepted and rejected triggers processed by the Reject Reset Type, a query only value determined by the trigger system. Any acquired waveform obtained by Accept or Reject Reset Types is positioned at the A-Event.

Group Trigger

Syntax

```
TRIGger:B:RESET:TOTALCOUNT?
```

Related Commands TRIGger:B:RESET:TYPe, TRIGger:B:RESET:RATIO?, TRIGger:B:RESET:RATIOENable, TRIGger:B:RESET:REJECTTIMEout, TRIGger:B:RESET:ACCEPTTIMEout, TRIGger:B:RESET:ACCEPTCOUNT?

Arguments None

Examples `TRIGGER:B:RESET:TOTALCOUNT?` query may return zero or any positive integer that represents the sum of the number of triggers accepted and rejected when the Accept or Reject Reset Types are active. Any acquired waveform obtained by Accept or Reject Reset Types is positioned at the A-Event.

TRIGger:B:RESET:TRANsition

This command sets or queries the type of transition required for a Transition trigger reset.

Group Trigger

Syntax

```
TRIGger:B:RESET:TRANsition {RISe|FALL}
TRIGger:B:RESET:TRANsition?
```

Related Commands TRIGger:B:RESET:SOUrce, TRIGger:B:RESET:THReshold, TRIGger:B:RESET:TYPe

Arguments

`RISe` indicates that a reset occurs when the trigger rises above the threshold level.

`FALL` indicates that a reset occurs when the trigger falls below the threshold level.

Examples

`TRIGGER:B:RESET:TRANSITION RISE` allows the reset to occur if the trigger rises above the threshold level.

`TRIGGER:B:RESET:TRANSITION?` might return `:TRIGGER:B:RESET:TRANSITION FALL`, indicating that the reset will occur if the trigger falls below the threshold level.

TRIGger:B:RESET:TYPe

This command sets or queries the type of A→B sequential trigger reset. If the B trigger reset is active, the reset criteria are part of the B triggering sequence. If the reset conditions defined by the reset type are not met, the instrument must start over searching for a new occurrence of the A event.

You must identify a trigger Source and Threshold for each reset type, except for the Timeout, Accept, and Reject trigger types.

NOTE. *Special considerations for Reset Type Accept: Reset Type Accept requires an accompanying Accept Timeout value. See TRIGger:B:RESET:ACCEPTTIMEout.*

NOTE. *Special considerations for Reset Type Reject: Reset Type Reject requires an accompanying Reject Timeout value. See TRIGger:B:RESET:REJECTTIMEout.*

NOTE. *The Reset Types Accept and Reject are only available on 70K Model D or later oscilloscopes.*

NOTE. *If a reset condition occurs, the reset criteria itself is reset and must start over.*

Group

Trigger

Syntax

```
TRIGger:B:RESET:TYPe
{NONe|TIMEOut|STATE|TRANsition|ACCepts|REJects}
TRIGger:B:RESET:TYPe?
```

Arguments

NONe defeats the trigger reset feature.

TIMEOut initiates a reset if the timeout conditions specified by TRIGger:B:RESET:TIMEOut are met.

STATE initiates a reset if the state conditions specified by TRIGger: B:RESET:SOUrce, TRIGger:B:RESET:STATE, and TRIGger:B:RESET:THReshold are met.

TRANsition initiates a reset if the transition conditions specified by TRIGger:B:RESET:SOUrce, TRIGger:B:RESET:TRANsition, and TRIGger:B:RESET:THReshold are met.

ACCepts initiates a trigger at the A-Event if the B-Event occurs before the Accept Timeout, as measured from the A- to the B-Event. That is, the B-Event Accepts (validates) the A- Event. Otherwise the oscilloscope does not trigger. The trigger location is at the A-Event.

REJects initiates a trigger if the Reject Timeout expires before the B-Event occurs, as measured from the A- to the B-Event. That is, reception of the B-Event Rejects (invalidates) the A-Event. Otherwise the oscilloscope does not trigger (if the B-Event occurs before the Reject Timeout). The trigger location is at the A-Event.

Examples

TRIGGER:B:RESET:TYPE NONE deactivates the sequential trigger reset.

TRIGGER:B:RESET:TYPE? This query might return :TRIGGER:B:RESET:TYPe TIMEOUT, indicating that the sequential trigger reset is active following a timeout.

TRIGger:B:SCAN:ADVANCEafter

This command sets or queries the B-Event Scan advance after value. Advance After advances the B-Event value after the specified number of acquisitions have occurred. The B-Events value is updated in a sequential, random, or toggle mode.

Group

Trigger

Syntax

```
TRIGger:B:SCAN:ADVANCEafter <NR1>
TRIGger:B:SCAN:ADVANCEafter?
```

Arguments

<NR1> is the value that advances the B-Event after the given number of acquisitions have occurred. The default value is 1 and ranges from 1 to 2e9 (2000000000).

Examples

`TRIGGER:B:SCAN:ADVANCEAFTER 1` with B-Event scan enabled, the B-Event advances after each acquisition.

`TRIGGER:B:SCAN:ADVANCEAFTER?` might return `:TRIGGER:B:SCAN:ADVANCEAFTER:2`, indicating that with B-Event scan enabled, the B-Event advances after two acquisitions.

TRIGger:B:SCAN:ENAble

This command sets or queries if the B-Event Scan is on or off. The B-Events value is updated in a sequential, random, or toggle Advance mode.

NOTE. *Not available on some models.*

Group Trigger

Syntax

```
TRIGger:B:SCAN:ENAble
TRIGger:B:SCAN:ENAble {ON|OFF}
TRIGger:B:SCAN:ENAble?
```

Arguments

`ON` with A->B Sequence mode set to Trig on nth Event, the B-Event count is determined by the other B Scan triggering parameters such as Start Event value, End Event value, Advance After value, and the Advance mode.

`OFF` with A->B Sequence mode set to Trig on nth Event, behaves in a conventional way.

Examples

`TRIGGER:B:SCAN:ENABLE ON` enables the B-Event Scan option.

`TRIGGER:B:SCAN:ENABLE?` might return `:TRIGGER:B:SCAN:ENABLE:OFF`, indicating that the B-Event Scan option is not enabled.

TRIGger:B:SCAN:ENDevent

This command sets or queries the end event value of B-Event Scan. The end event value is greater than or equal to the start event value.

Group Trigger

Syntax

```
TRIGger:B:SCAN:ENDevent <NR1>
TRIGger:B:SCAN:ENDevent?
```

Related Commands TRIGger:B:SCAN:STARTevent

Arguments <NR1> is the end event value of B-Event Scan. The default value is 8 and ranges from 1 to 2e9.

Examples TRIGGER:B:SCAN:ENDEVENT 8 sets the end event value of B-Event Scan to 8. At this point the B-Event value is set to the start event value again.

TRIGGER:B:SCAN:ENDEVENT? might return :TRIGGER:B:SCAN:ENDEVENT 10, indicating that the end event value of B-Event scan is set to 10.

TRIGger:B:SCAN:MODE

This command sets or queries the sequence advance mode of B-Event Scan.

Group Trigger

Syntax
```
TRIGger:B:SCAN:MODE {SEQUENTIAL|RANDOM|TOGGLE}
TRIGger:B:SCAN:MODE?
```

Arguments SEQUENTIAL, where the B-Event Scan is initialized at start event. This value is incremented after each set of Advance After triggers have occurred, until the B-Event exceeds the end event value. Then the B-Event count is again set to the start event value.

RANDOM, where the B-Event value is changed to a random value such that start event value <= BEvent <= end event value), and after Advance After triggers have occurred.

TOGGLE, where the B-Event count is first set to the start event value and then changed to the end event value after additional Advance After triggers have occurred, and then B-Event is set back to the start event value.

Examples TRIGGER:B:SCAN:MODE SEQUENTIAL sets the advance mode of the B-Event Scan to sequential.

TRIGGER:B:SCAN:MODE? might return :TRIGGER:B:SCAN:MODE SEQUENTIAL, indicating that the advance mode of the B-Event Scan is set to sequential.

TRIGger:B:SCAN:STARTevent

This command sets or queries the start event value of B-Event Scan.

Group Trigger

Syntax
```
TRIGger:B:SCAN:STARTevent <NR1>
TRIGger:B:SCAN:STARTevent?
```

Related Commands TRIGger:B:SCAN:ENDevent

Arguments <NR1> is the start event value of B-Event Scan. The default value is 1 and ranges from 1 to 2e9.

Examples TRIGGER:B:SCAN:STARTEVENT 1 initializes the start event value of B-Event Scan at 1.

TRIGGER:B:SCAN:STARTEVENT? might return :TRIGGER:B:SCAN:STARTEVENT 2, indicating that the start event value of B-Event scan is initialized at 2.

TRIGger:B:STATE

This command sets or queries the state of B trigger activity. If the B trigger state is on, the B trigger is part of the triggering sequence. If the B trigger state is off, then only the A trigger causes the trigger event.

Group Trigger

Syntax
```
TRIGger:B:STATE {ON|OFF|<NR1>}
TRIGger:B:STATE?
```

Related Commands TRIGger:A:MODe

Arguments ON indicates that the B trigger is active and causes trigger events with the A trigger.

OFF indicates that only the A trigger causes trigger events.

<NR1>

A 0 turns off the B trigger; any other value activates the B trigger.

Examples TRIGGER:B:STATE ON sets the B trigger to active, making it capable of causing trigger events.

TRIGGER:B:STATE? might return :TRIGGER:B:STATE 0, indicating that the B trigger is inactive and that only the A trigger causes trigger events.

TRIGger:B:TIMe

This command sets or queries B trigger delay time. The B Trigger time applies only if TRIGger:B:BY is set to TIMe. This command is equivalent to selecting B Event (Delayed) Trigger Setup from the Trig menu, choosing the A®B Seq tab, and setting Trig Delay.

Group Trigger

Syntax

```
TRIGger:B:TIMe <NR3>
TRIGger:B:TIMe?
```

Related Commands TRIGger:B:BY, TRIGger:B:EVENTS:COUNt TRIGger:B:STATE

Arguments <NR3> is the B trigger delay time in seconds.

Examples TRIGGER:B:TIME 4E-6 sets the B trigger delay time to 4 µs.

TRIGGER:B:TIME? might return :TRIGGER:B:TIME 16.0000E-9, indicating that the B trigger time is set to 16 ns.

TRIGger:ENHanced

This command sets or queries the state of trigger position enhancement. When on, the instrument improves the trigger positioning to more closely match the acquired data. This is equivalent to selecting Enhanced Triggering from the Trigger Mode menu.

Group Trigger

Syntax

```
TRIGger:ENHanced {<NR1>|OFF|ON}
TRIGger:ENHanced?
```

Arguments <NR1> = 0 disables trigger position enhancement, any other value enables trigger position enhancement.

OFF disables trigger position enhancement.

ON enables trigger position enhancement.

Examples

TRIGGER:ENHANCED ON enables trigger position enhancement.

TRIGGER:ENHANCED? might return :TRIGGER:ENHANCED 0, indicating that trigger position enhancement is disabled.

TRIGger:EQUation<1..1>

Sets the Visual Trigger Equation string, which defines the behavior of the areas.

There can be only one equation, which can be up to 128 characters. You can enter the command as TRIGGER:EQUATION or TRIGGER:EQUATION1.

Each area is assigned a single source (analog channel 1, 2, 3, or 4). Any analog channel can be used as the source for one or more areas.

Some basic equations are:

- (C1 IN A1) — The channel 1 waveform must intersect with Area 1.
- (C2 OUT A2) — The channel 2 waveform must NOT intersect with Area 2.
- ((C1 IN A1) and (C2 OUT A2)) — Combines the previous two examples.

Group

Trigger

Syntax

```
TRIGger:EQUation<1..1> <string>
TRIGger:EQUation<1..1>?
```

Related Commands

VISual:ENAble, TRIGger:SHOWEQuation

Arguments

<string> is the equation that you want to use for visual triggering. This equation is shown on the oscilloscope screen when visual triggering is enabled.

Examples

TRIGger:EQUATION1 "((C1 IN A1) and (C2 OUT A2))" provides the equation that will control how areas 1 and 2 are used for visual triggering.

TRIGger:EQUATION? might return :TRIGger:EQUATION "((C1 IN A2) and (C2 OUT A3))", providing the equation that will control how areas 2 and 3 are used for visual triggering.

TRIGger:LVLSrcpreference

This command sets or queries the dependent source/level trigger feature. SRCIndependent is the default value.

Group Trigger

Syntax
```
TRIGger:LVLSrcpreference {SRCDependent|SRCIndependent}
TRIGger:LVLSrcpreference?
```

Related Commands TRIGger:{A|B}:LEVel:CH<x>,

Arguments `SRCDependent` sets the level of all trigger sources to the value you currently select, regardless of the last value selected.

`SRCIndependent` sets each trigger source to the level you are currently selecting.

Examples `TRIGGER:LVLSRCPREFERENCE SRCIINDEPENDENT` allows you to set the level of each source to the value you are currently selecting.

`TRIGGER:LVLSRCPREFERENCE?` might return `:TRIGGER:LVLSRCPREFERENCE SRCDEPENDENT`, indicating that the level of all sources changes to the value you currently select.

TRIGger:QUALification:BUS:FORMat

This command sets or queries the format of the bit pattern for the bus qualification. The command is available only when the Trigger Type is set to Glitch, Runt, Setup/Hold, Timeout, Transition, Width, or Window and the trigger qualification is set to Occurs And.

Group Trigger

Syntax
```
TRIGger:QUALification:BUS:FORMat {BINary|HEX|SYMBolic}
TRIGger:QUALification:BUS:FORMat?
```

Related Commands TRIGger:QUALification:BUS:SOUrce, TRIGger:QUALification:BUS:VALue

Arguments `BINary` specifies the pattern format as binary.

`HEX` specifies the pattern format as hexadecimal.

SYMBolic specifies the pattern format as symbolic.

Examples

TRIGGER:QUALIFICATION:BUS:FORMAT HEX sets the bit pattern format to hexadecimal.

TRIGGER:QUALIFICATION:BUS:FORMAT? might return :TRIGGER:QUALIFICATION:BUS:FORMAT BINARY indicating that the bit pattern format to binary.

TRIGger:QUALification:BUS:SOUrce

This command sets or queries the source for the bus qualification. The command is available only when the Trigger Type is set to Glitch, Runt, Setup/Hold, Timeout, Transition, Width, or Window and the trigger qualification is set to Occurs And.

Group

Trigger

Syntax

```
TRIGger:QUALification:BUS:SOUrce
{B0|B1|B2|B3|B4|B5|B6|B7|B8|B9|B10|B11|B12|B13|B14|B15|B16}
TRIGger:QUALification:BUS:SOUrce?
```

Related Commands

TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:VALue

Arguments

B0–B16 specifies the bus source from B0 to B16.

Examples

TRIGGER:QUALIFICATION:BUS:SOURCE B1 sets the source for bus qualification to B1.

TRIGGER:QUALIFICATION:BUS:SOURCE? might return :TRIGGER:QUALIFICATION:BUS:SOURCE B5 indicating that the source for bus qualification is set to B5.

TRIGger:QUALification:BUS:VALue

This command sets or queries the bit pattern for the bus qualification. The command is available only when the Trigger Type is set to Glitch, Runt, Setup/Hold, Timeout, Transition, Width, or Window and the trigger qualification is set to Occurs And. The pattern assigned is in the order D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0 Ch1 Ch2 Ch3 Ch4.

Group

Trigger

Syntax

```
TRIGger:QUALification:BUS:VALue <String>
TRIGger:QUALification:BUS:VALue?
```

Related Commands TRIGger:QUALification:BUS:FORMat, TRIGger:QUALification:BUS:SOUrce

Arguments <String> sets the bus value.

Examples TRIGGER:QUALIFICATION:BUS:VALUE "1 XXXX" sets the bus value to 1xxx (a logic high on D0 and CH1 through CH4 to "don't care" value).

TRIGGER:QUALIFICATION:BUS:VALUE? might return :TRIGGER:QUALIFICATION:BUS:VALUE "01XXXX" indicating that the value for bus qualification is set to 01XXXX (a logic low is expected on D1, logic high on D0, and a "don't care" value on CH1 through CH4).

TRIGger:SENSITivity

This command set or queries the state of trigger sensitivity. When on, the instrument calculates a rolling average for trigger position enhancement to suppress trigger jitter in noisy signals. This is equivalent to selecting Higher Sensitivity for Noisy Signals from the Trigger Mode menu. This is available only when TRIGger:ENHanced or the equivalent check box control is also on.

Group Trigger

Syntax

```
TRIGger:SENSITivity {<NR1>|OFF|ON}
TRIGger:SENSITivity?
```

Related Commands TRIGger:ENHanced

Arguments <NR1> = 0 disables rolling averaging, any other value enables rolling averaging for trigger position enhancement.

OFF disables rolling averaging for trigger position enhancement.

ON enables rolling averaging for trigger position enhancement.

Examples TRIGGER:SENSITIVITY ON enables the trigger sensitivity feature when TRIGger:ENHanced is enabled.

TRIGGER:SENSITIVITY? might return :TRIGGER:SENSITIVITY 1, indicating that the TRIGGER:SENSITIVITY feature is enabled.

TRIGger:SHOWEQuation

Causes the Visual Trigger Equation to be displayed in the upper left portion of the oscilloscope graticule.

Group Trigger

Syntax
```
TRIGger:SHOWEQuation {ON|OFF}
TRIGger:SHOWEQuation?
```

Related Commands TRIGger:EQUation<1..1>

Arguments `ON` causes the equation to appear on screen.

`Off` hides the equation.

Examples `TRIGGER:SHOWEQUATION ON` displays the equation.

`TRIGGER:SHOWEQUATION?` might return `:TRIGGER:SHOWEQUATION OFF`, indicating that the equation is not set to display.

TRIGger:STATE? (Query Only)

This query-only command returns the current state of the triggering system. This command is equivalent to viewing the trigger status LEDs on the instrument front panel, and updates on a periodic, timed basis. For immediate, synchronous, querying of the trigger status, see TRIGger:{A|B}:READY?

Group Trigger

Syntax
```
TRIGger:STATE?
```

Related Commands TRIGger:A:MODe, TRIGger:{A|B}:READY?

Returns `ARMED` indicates that the instrument is acquiring pretrigger information.

`AUTO` indicates that the instrument is in the automatic mode and acquires data even in the absence of a trigger.

`DPO` indicates that the instrument is in DPO mode.

PARTIAL indicates that the A trigger has occurred and the instrument is waiting for the B trigger to occur.

READY indicates that all pretrigger information is acquired and that the instrument is ready to accept a trigger.

SAVE indicates that the instrument is in save mode and is not acquiring data.

TRIGGER indicates that the instrument 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)

This query-only command tests (self-test) the GPIB interface and returns a 0.

Group Miscellaneous

Syntax *TST?

Examples *TST? always returns 0.

UNLock (No Query Form)

This command (no query form) unlocks the front panel. The command is equivalent to LOCk NONe.

NOTE. *If the instrument is in the Remote With Lockout State (RWLS), the UNLock command has no effect. For more information, see the ANSI-IEEE Std 488.1-1987 Standard Digital Interface for Programmable Instrumentation, section 2.8.3 on RL State Descriptions.*

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:PRODUCTID:DECimal? (Query Only)

This command queries the USBTMC product ID in decimal. The product ID numbers vary for each instrument family and model.

Group Miscellaneous

Syntax USBTMC:PRODUCTID:DECimal?

Examples USBTMC:PRODUCTID:DECIMAL? might return :USBTMC:PRODUCTID:DECIMAL 0, indicating that the USBTMC product ID in decimal is 0.

USBTMC:PRODUCTID:HEXadecimal? (Query Only)

This command queries the USBTMC product ID in hexadecimal. The product ID numbers vary for each instrument family and model.

Group Miscellaneous

Syntax USBTMC:PRODUCTID:HEXadecimal?

Examples USBTMC:PRODUCTID:HEXADECIMAL? might return :USBTMC:PRODUCTID:HEXADECIMAL 0x0000, indicating that the USBTMC product ID in hexadecimal is 0x0000.

USBTMC:SERIALnumber? (Query Only)

This command queries the instrument serial number.

Group Miscellaneous

Syntax USBTMC:SERIALnumber?

Examples `USBTMC:SERIALNUMBER?` might return `:USBTMC:SERIALNUMBER Q100001`, indicating that the instrument serial number is Q100001.

USBTMC:VENDORID:DECimal? (Query Only)

This command queries the USBTMC product ID in decimal. The product ID numbers vary for each instrument family and model.

Group Miscellaneous

Syntax `USBTMC:VENDORID:DECimal?`

Examples `USBTMC:VENDORID:DECIMAL?` might return `:USBTMC:VENDORID:DECIMAL 0`, indicating that the USBTMC product ID in decimal is 0.

USBTMC:VENDORID:HEXadecimal? (Query Only)

This command queries the USBTMC product ID in hexadecimal. The product ID numbers vary for each instrument family and model.

Group Miscellaneous

Syntax `USBTMC:VENDORID:HEXadecimal?`

Examples `USBTMC:VENDORID:HEXADECIMAL?` might return `:USBTMC:VENDORID:HEXADECIMAL 0x0000`, indicating that the USBTMC product ID in hexadecimal is 0x0000.

VERBose

This command sets or queries 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). However, this command does make a corresponding change in the Response Header Enable State of the opposite interface (physical or virtual GPIB interface). Refer to Introduction for more information.*

Group Miscellaneous

Syntax `VERBose {<NR1>|OFF|ON}`

Related Commands HEADer, *LRN?, SET?

Arguments `<NR1>`

`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.

A 0 returns minimum-length keywords for applicable setting queries; any other value returns full-length keywords.

Examples `VERBOSE ON` sets the Verbose state to true and return the full length keyword for the applicable setting queries.

`VERBOSE?` might return `:VERBOSE 0FF`, indicating that the Verbose state is set to false and return the minimum-length keywords for the applicable setting queries.

VISual? (Query Only)

This query-only command returns the settings for each visual trigger area.

Group Trigger

Syntax `VISual?`

Examples `VISUAL:ENABLE` might return `:VISUAL:ENABLE 1;AREA1:LABEL "";DISPLAY 1;SHOWLOGIC 1;ROTATION 0.0000;SOURCE1 CH1;OPERATION1 IN;XPOSITION -6.0000E-6;YPOSITION 719.0000E-3;WIDTH 10.8000E-6;HEIGHT 1.2540;:VISUAL:AREA2:LABEL "";DISPLAY 1;SHOWLOGIC 1;ROTATION 0.0000;SOURCE1 CH1;OPERATION1 IN`, listing the settings for visual areas 1 and 2.

VISual:AREACOLOr

This command sets or queries the colors used by visual trigger areas.

Group Trigger

Syntax
```
VISual:AREACOLOr {DEFAULT|INHERIT}
VISual:AREACOLOr?
```

Arguments `DEFAULT` sets visual trigger areas to use the default blue color.

`INHERIT` sets visual trigger areas to inherit the color of the channel.

Examples `VISUAL:AREACOLOR` inherit causes visual trigger areas to inherit the color of the channel.

`VISUAL:AREACOLOR?` might return `:VISUAL:AREACOLOR DEFAULT`, indicating that visual trigger areas will use the default blue color.

VISual:AREA<x>? (Query Only)

This query-only command returns the settings for the specified visual trigger area. The area is specified by x. The value of x can range from 1 through 8.

Group Trigger

Syntax
```
VISual:AREA<x>?
```

Examples `VISUAL:AREA1:LABEL?` might return `:VISUAL:AREA1:LABEL "";DISPLAY 1;SHOWLOGIC 1;ROTATION 0.0000;SOURCE1 CH1;OPERATION1 IN;XPOSITION -6.0000E-6;YPOSITION 719.0000E-3;WIDTH 10.8000E-6;HEIGHT 1.2540`, listing the settings for visual area 1.

VISual:AREA<x>:DISplay

Causes the areas to be hidden or displayed on the oscilloscope screen. Queries the status of the area display.

All areas are set On or Off together. The <x> in the command is reserved for future use.

Group Trigger

Syntax
```
VISual:AREA<x>:DISplay {ON|OFF}
VISual:AREA<x>:DISplay?
```

Related Commands VISual:AREA<x>:SHOWLOGic

Arguments `ON` causes the areas to appear on screen.

`Off` hides the areas.

Examples `VISual:AREA1:DISplay ON` turns on the graphical display of the areas on the oscilloscope screen.

`VISual:AREA1:DISplay?` might return `:VISUAL:AREA1:DISPLAY OFF`, indicating that the areas are not displayed on the screen.

VISual:AREA<x>:FLIP (No Query Form)

This command flips the specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax `VISual:AREA<x>:FLIP {HORIZONTAL|VERTICAL}`

Arguments `HORIZONTAL` specifies to flip the area in the horizontal direction.

`VERTICAL` specifies to flip the area in the vertical direction.

Examples `VISual:AREA1:FLIP VERTICAL` flips visual trigger area 1 vertically.

VISual:AREA<x>:HEIGHT

This command sets or queries the height of the specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax
```
VISual:AREA<x>:HEIGHT <nr3>
VISual:AREA<x>:HEIGHT?
```

Arguments `<NR3>` specifies the height of the specified visual trigger area.

Examples `VISual:AREA1:HEIGHT 1.0` sets the height of visual trigger area 1 to 1.0 V.

`VISual:AREA1:HEIGHT?` might return `:VISUAL:AREA1:HEIGHT 1.0480`, indicating the that height of visual trigger area 1 is 1.048 V.

VISual:AREA<x>:LABel

This command sets or queries the label of the specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax
```
VISual:AREA<x>:LABel <string>
VISual:AREA<x>:LABel?
```

Arguments `<string>` specifies the label for the area.

Examples `VISUAL:AREA1:LABEL "Glitch 1"` labels visual trigger area 1 as Glitch 1.

`:VISUAL:AREA1:LABEL?` might return `:VISUAL:AREA1:LABEL ""`, indicating that area 1 is not labeled.

VISual:AREA<x>:OPERATION<1..1>

Sets or queries the operation of the area specified by <x>.

You can enter the command as `VISUAL:AREA<x>:OPERATION` or `VISUAL:AREA<x>:OPERATION1`.

Group Trigger

Syntax
```
VISual:AREA<x>:OPERATION<1..1> {IN|OUT}
VISual:AREA<x>:OPERATION<1..1>?
```

Related Commands VISual:AREA<x>:SOURCE<1..1>

Arguments `IN` specifies that the selected source must pass through the specified area.

`OUT` specifies that the selected source must NOT pass through the specified area.

Examples `VISual:AREA1:OPERATION OUT` specifies that the selected source must NOT pass through area 1.

`VISual:AREA5:OPERATION?` might return `:VISUAL:AREA5:OPERATION IN`, indicating that the selected source must pass through area 5.

VISual:AREA<x>:ROTAtion

This command sets or queries the rotation angle of the specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax
```
VISual:AREA<x>:ROTAtion <nr3>
VISual:AREA<x>:ROTAtion?
```

Arguments `<NR3>` specifies the rotation angle of the specified visual trigger area.

Examples `VISual:AREA1:ROTAtion 45` rotates visual trigger area 1 by 45 degrees.

`VISual:AREA1:ROTAtion?` might return `:VISUAL:AREA1:ROTATION 0.0000`, indicating that visual trigger area is not rotated.

VISual:AREA<x>:SHAPE

Selects the initial shape of area <x>.

Group Trigger

Syntax
```
VISual:AREA<x>:SHAPE {HEXAgon|RECTangle|TRAPezoid|TRIAngle}
VISual:AREA<x>:SHAPE?
```

Related Commands VISual:AREA<x>:DISplay, VISual:AREA<x>:VERTICES

Arguments `HEXAGON` sets the initial shape of the selected area to a hexagon.

`RECTANGLE` sets the initial shape of the selected area to a rectangle.

`TRAPEZOID` sets the initial shape of the selected area to a trapezoid.

`TRIANGLE` sets the initial shape of the selected area to a triangle.

Examples `VISual:AREA2:SHAPE RECTANGLE` sets the initial shape of area 2 to a rectangle.

`VISual:AREA3:SHAPE?` might return `:VISUAL:AREA3:SHAPE HEXAGON`, indicating that the shape of area 3 is a hexagon. This can be used with an eye diagram.

VISual:AREA<x>:SHOWLOGic

Causes area Source and Operation indicators to be hidden or displayed on screen. Queries the status of this feature.

All areas are turned on and off together. The <x> is reserved for future use.

Group Trigger

Syntax
```
VISual:AREA<x>:SHOWLOGic {ON|OFF}
VISual:AREA<x>:SHOWLOGic?
```

Related Commands VISual:AREA<x>:DISplay

Arguments `ON` shows the definition (source, operation, and area label) of each area within the graphical display of that area.

`OFF` removes the definitions from the display.

Examples `VISual:AREA:SHOWLOGic ON` sets the oscilloscope to display the definitions for the areas.

`VISual:AREA:SHAPE?` might return `:VISUAL:AREA:SHOWLOGIC OFF`, indicating that the area definitions are not displayed.

VISual:AREA<x>:SOURCE<1..1>

Selects or queries the signal source for the area <x>. The source can only be an analog channel.

You can enter the command as `VISUAL:AREA<x>:SOURCE` or `VISUAL:AREA<x>:SOURCE1`.

Group Trigger

Syntax
```
VISual:AREA<x>:SOURCE<1..1> {WFMCH1|WFMCH2|WFMCH3|WFMCH4}
VISual:AREA<x>:SOURCE<1..1>?
```

Related Commands VISual:AREA<x>:OPERATION<1..1>

Arguments `WFMCH1` indicates that the analog channel 1 waveform is used as the signal source for the area <x>. The other three channels have the same syntax.

Examples `VISual:AREA4:SOURCE WFMCH1` indicates that the signal source for area 4 is the channel 1 waveform.

`VISual:AREA5:SOURCE?` might return `:VISual:AREA5:SOURCE WFMCH4`, indicating that the signal source for area 5 is the channel 4 waveform.

VISual:AREA<x>:VERTICES

Sets or queries the polygon vertex x and y coordinate values for an area. You must set vertex values in pairs; you can set 3, 4, or 6 pairs (no pentagons are allowed).

Group Trigger

Syntax

```
VISual:AREA<x>:VERTICES {<NR3>,<NR3>,<NR3>,<NR3>,<NR3>,<NR3>
[,<NR3>,<NR3>,<NR3>,<NR3>[,<NR3>,<NR3>]]}
VISual:AREA<x>:VERTICES?
```

Related Commands VISual:AREA<x>:DISplay, VISual:AREA<x>:SHAPE

Arguments <NR3> is entered in pairs; the first value is x (time) and the second value is y (amplitude). The units of measure are determined by the oscilloscope settings; the amplitude is generally in volts.

Examples `VISual:AREA:3:VERTICES ”-1.0e-9, -1.0e-3, -1.0e-9, 1.0e-3, 1.0e-9, 1.0e-3, 1.0e-9, -1.0e-3”` defines area 3 as a rectangle that is 200 ns wide by 200 mV high, centered on the display.

`VISual:AREA:3:VERTICES?` might return`”-1.0e-9, -1.0e-3, -1.0e-9, 1.0e-3, 1.0e-9, 1.0e-3, 1.0e-9, -1.0e-3”`, indicating that area 3 is defined as in the preceding example.

VISual:AREA<x>:WIDTH

This command sets or queries the width of the specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax
```
VISual:AREA<x>:WIDTH <nr3>
VISual:AREA<x>:WIDTH?
```

Arguments <NR3> specifies the width of the visual trigger area.

Examples VISual:AREA<x>:WIDTH 0.0000045 sets the width to 4.5 µs.

VISual:AREA<x>:WIDTH? might return :VISUAL:AREA1:WIDTH 6.8000E-6, indicating the width of the area is set to 6.8 µs.

VISual:AREA<x>:XPOSition

This command sets or queries horizontal position of specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax
```
VISual:AREA<x>:XPOSition <nr3>
VISual:AREA<x>:XPOSition?
```

Arguments <NR3> specifies the horizontal position of the center of the visual trigger area.

Examples VISual:AREA1:XPOSition -6.9 sets the horizontal center of the visual trigger area to –6.9 µs from the trigger point.

VISual:AREA1:XPOSition? might return :VISUAL:AREA1:XPOSITION -5.1000E-6, indicating that the x position of the area is –5.1 µs from the trigger point.

VISual:AREA<x>:YPOSition

This command sets or queries vertical position of the specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax
```
VISual:AREA<x>:YPOSition <nr3>
VISual:AREA<x>:YPOSition?
```

Arguments <NR3> specifies the vertical position of the center of the visual trigger area.

Examples VISual:AREA1:YPOSition 0.5 sets the vertical position of the area to 500 mV.

VISual:AREA1:YPOSition? might return :VISUAL:AREA1:YPOSITION 532.0000E-3, indicating that the vertical position of the area is set to 532 mV.

VISual:ASPECTratio

This command sets or queries the aspect ratio setting of the visual trigger system.

Group Trigger

Syntax
```
VISual:ASPECTratio {<NR1>|OFF|ON}
VISual:ASPECTratio?
```

Arguments <NR1> = 0 disables the function; any other value enables it.

OFF disables the function.

ON enables keeping the aspect ratio constant.

Examples VISUAL:ASPECTRATIO ON enables keeping the aspect ratio constant.

VISUAL:ASPECTRATIO? might return :VISUAL:ASPECTRATIO 0, indicating that the aspect ratio will not be kept constant.

VISual:DELETEAREA (No Query Form)

This command deletes the specified visual trigger area. The area is specified by x. The value of x can range from 1 to 8.

Group Trigger

Syntax VISual:DELETEAREA <1–8>

Examples VISUAL:DELETEAREA 3 deletes visual trigger area 3.

VISual:ENAble

Enables or disables the visual triggering. Queries the status (on or off) of the visual triggering. This does not refer to the option key.

Group Trigger

Syntax
```
VISual:ENAble {ON|OFF}
VISual:ENAble?
```

Arguments `ON` enables the visual triggering feature.

`OFF` disables the visual triggering feature.

Examples `VISUAL:ENABLEON` turns on the visual triggering.

`VISUAL:ENABLE?` might return `:VISUAL:ENABLE OFF`, indicating that visual triggering will not occur.

VISual:FILE:RECALL (No Query Form)

Recalls a file of Visual Trigger Parameters, updating the database appropriately. The file is stored in the form of programmer interface commands, which are human readable.

Group Trigger

Syntax
```
VISual:FILE:RECALL <string>
```

Related Commands VISual:FILE:SAVE

Arguments <string> is the complete path and name of the visual trigger parameters file you want to recall. This can be on a local or network drive.

Examples
```
VISUAL:FILE:RECALL
```

`C:\users\username\Tektronix\TekScope\VisualSetups\visualsetup3.txt`" provides the name and location of the visual trigger parameters file that will be recalled.

VISual:FILE:SAVE (No Query Form)

Saves only the Visual Trigger Parameters, to a file that you name in the <string>. The file is stored in the form of programmer interface commands, which are human readable.

Group Trigger

Syntax `VISual:FILE:SAVE <string>`

Related Commands VISual:FILE:RECALL

Arguments `<string>` is the complete path and file name where you want to store the visual trigger parameters. This can be a local or network drive.

Examples `:VISUAL:FILE:SAVE "C:\users\username\Tektronix\TekScope\VisualSetups\mysetup.txt"` sets that path and file name as the place to store the visual trigger parameters.

*WAI (No Query Form)

The *WAI (Wait) command (no query form) prevents the instrument 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 instrument with your application program. For more information, refer to Synchronization Methods.

Group Status and Error

Syntax `*WAI`

Related Commands BUSY?, *OPC

Examples `*WAI` prevents the instrument from executing any further commands or queries until all pending commands that generate an OPC message are complete.

WAVFrm? (Query Only)

This query-only command 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.

Group Waveform Transfer

Syntax `WAVFrm?`

Related Commands CURVe, DATa:SOUrce, WFMOutpre?

Examples `WAVFRM?` might return the waveform data as: `:WFMOUTPRE:BIT_NR 8;BN_FMT RI;BYT_NR 1; BYT_OR MSB;ENCDG ASC;NR_PT 500;PT_FMT Y; PT_ORDER LINEAR;PT_OFF 0;XINCR 400.0000E-12; XZERO 0.0000;XUNIT "s";YMULT 4.0000E-3; YOFF 0.0000;YZERO 0.0000;YUNIT "V"; WFID "Ch1,DC coupling, 100.0mV/div, 200.0ns/div, 5000 points,Sampl mode"; :CURVE51,50,51,48,51,48,50,49,51,49,51,48,51,48,51,49, 50,49,50,48,49,49,52,49,49,50,50,48,50,49,49, 49,49,49,50,47,49,47,50,48,49,48,50,48,49,47, 49,48,51,48,49,47,50,48,50,47,51,47,49,48,48, 48,50,46,50,46,48,45,48,47,49,47,49,48,49,48, 49,45,49,47,48,46,48,48,49,45,49,45,47,46,46, 46,49,47,48,46,48,46,48,47,47,46,47,47,48,46, 48,45,48,46,47,46,47,47,46,45,46,45,47,47,47, 46,46,44,47,45,45,45,46,43,46,45,46,44,46,45, 48,45,45,44,46,45,46,45,45,45,45,43,45,44,46, 44,47,44,44,44,45,43,44,44,45,44,46,43,46,43, 45,44,45,44,45,41,44,43,45,42,44,43,44,43,44, 42,43,42,44,41,44,41,44,42,43,42,43,42,45,42, 43,41,42,41,43,42,44,42,43,42,43,40,42,40,42, 40,42,41,43,41,42,41,42,39,41,41,42,42,43,41, 42,39,41,39,41,40,41,40,41,39,41,38,41,39,41, 38,42,39,41,37,40,38,40,38,39,38,39,38,41,39, 39,38,40,37,38,37,38,38,39,38,40,36,39,37,39, 37,39,37,39,36,40,38,38,35,38,36,38,37,39,37, 38,37,38,37,37,36,38,35,38,36,37,36,38,36,38, 35,39,34,36,36,36,35,38,35,36,34,37,34,37,35, 37,36,37,35,37,35,36,34,36,33,36,35,36,34,36, 33,34,33,34,34,34,34,36,33,36,34,36,33,34,33, 34,32,34,32,34,33,33,31,33,32,34,32,33,32,32, 31,33,31,31,32,33,31,32,31,32,30,32,30,32,31,`

```
31,30,31,30,31,29,31,29,32,31,31,30,30,29,30,
29,30,30,31,29,30,28,30,29,31,28,30,28,30,29,
29,27,29,28,29,27,30,27,30,26,28,28,29,27,28,
27,28,27,28,28,28,27,28,27,28,26,28,26,28,26,
26,25,27,25,28,27,28,26,25,25,26,25,25,25,25,
25,26,25,27,25,25,24,25,25,26,24,26,24,24,23,
25,24,24,22,25,23,24,23,24,22,24,22,23,22,25,
24,24,21,23,22,22,21,24,22,22,22,22,21,22,20, 23,21,21,20.
```

WAVFRMStream? (Query Only)

This query only command returns WFMQUTPRE? and CURVESTREAM? data for the waveforms specified by the DATASOURCE command. This command is similar to sending both WFMOUTPRE? and CURVESTREAM?, with the additional provision that each CURVESTREAM response to WAVFRMS? has a WFMOUTPRE response prepended to it. This helps guarantee a continuous synchronized preamble and curve.

Group Waveform Transfer

Syntax WAVFRMStream?

Returns WFMOUTPRE? and CURVESTREAM? query responses.

Examples Single data source (1 channel, record length 1000). WAVFRMSTREAM? might return

```
:WFMOUTPRE <WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>

:WFMOUTPRE <WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>

. . .
```

Multiple data source (record length 1000). WAVFRMSTREAM? might return (wf<x> denotes one of each waveform source specified by :DATA:SOURCE WF1,WF2,WF3)

```
:WFMOUTPRE <WF1 WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>

:WFMOUTPRE <WF2 WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>

:WFMOUTPRE <WF3 WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>
```

```
:WFMOUTPRE <WF1 WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>

:WFMOUTPRE <WF2 WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>

:WFMOUTPRE <WF3 WFMOUTPRE branch query>;:CURVE #41000<binary
data>;<newline>

. . .
```

WFMInpre? (Query Only)

This query-only command returns the waveform formatting specification 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_FMTRI;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

This command sets or queries the number of bits per binary waveform point for the waveform, as specified by the DATa:DESTination command. This specification is meaningful only when WFMInpre:ENCdg is set to BIN.

Group Waveform Transfer

Syntax

```
WFMInpre:BIT_Nr <NR1>
WFMInpre:BIT_Nr?
```

Related Commands DATa:DESTination, WFMInpre:ENCdg, WFMInpre:BYT_Nr, WFMOutpre:BIT_Nr

Arguments <NR1> number of bits per data point can be 8, 16 (RI, RP) or 32 (FP).

Examples WFMINPRE:BIT_NR 16 sets the number of bits per waveform point to 16, for incoming RI and RP binary format data.

WFMINPRE:BIT_NR? might return :WFMINPRE:BIT_NR 8, indicating that incoming RI or RP binary format data uses 8 bits per waveform point.

WFMInpre:BN_Fmt

This command sets or queries the format of binary data for incoming waveforms.

Group Waveform Transfer

Syntax
```
WFMInpre:BN_Fmt {RI|RP|FP}
WFMInpre:BN_Fmt?
```

Related Commands WFMOutpre:BN_Fmt

Arguments RI specifies signed integer data point representation.

RP specifies positive integer data point representation.

FP specifies single-precision binary floating point representation.

Examples WFMINPRE:BN_FMT FP specifies that incoming data will be interpreted as single-precision binary floating point numbers.

WFMINPRE:BN_FMT? might return :WFMINPRE:BN_FMT RI, indicating that the incoming data is currently interpreted as signed integers.

WFMInpre:BYT_Nr

This command sets or queries the binary field data width for the first ordered waveform, as specified by the DATa:DESTination command. This specification is only meaningful when WFMInpre:ENCdg is set to BIN and WFMInpre:BN_Fmt is set to either RI or RP.

Group Waveform Transfer

Syntax

```
WFMInpre:BYT_Nr <NR1>
WFMInpre:BYT_Nr?
```

Related Commands DATa:DESTination, WFMInpre:BN_Fmt, WFMInpre:ENCdg, WFMInpre:BIT_Nr, WFMOutpre:BYT_Nr

Arguments <NR1> is the number of bytes per data point and can be 1, 2 (RI, RP) or 4 (FP).

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

This command sets or queries which byte of binary waveform data is transmitted first for incoming waveform data when data points require more than one byte. This specification only has meaning when WFMInpre:ENCdg is set to BIN and WFMInpre:BYT_Nr is greater than 1.

Group Waveform Transfer

Syntax

```
WFMInpre:BYT_Or {LSB|MSB}
WFMInpre:BYT_Or?
```

Related Commands WFMInpre:ENCdg, WFMInpre:BYT_Nr, WFMOutpre:BYT_Or

Arguments LSB specifies that the least significant byte will be transmitted first.

MSB specifies that the most significant byte will be transmitted first.

Examples WFMINPRE:BYT_OR MSB sets the most significant incoming byte of incoming waveform data to be transmitted first.

WFMINPRE:BYT_OR? might return :WFMINPRE:BYT_OR LSB, indicating that the least significant incoming CURVe data byte will be transmitted first.

WFMInpre:ENCdg

This command sets or queries the type of encoding for incoming waveform data.

Group Waveform Transfer

Syntax
```
WFMInpre:ENCdg {ASCii|BINary}
WFMInpre:ENCdg?
```

Related Commands WFMOutpre:ENCdg

Arguments `ASCii` specifies that the incoming data is in `ASCii` II 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 ASCii` sets the format of incoming waveform data to ASCII format.

`WFMINPRE:ENCDG?` might return `:WFMINPRE:ENCDG BINary`, indicating that the incoming waveform data is in binary format.

WFMInpre:NR_FR? (Query Only)

This query returns the number of frames to be transferred to the instrument with the CURVe command.

Group Waveform Transfer

Syntax
```
WFMInpre:NR_FR?
```

Related Commands CURVe, DATa, DATa:STARt, DATa:STOP, SAVe:WAVEform, SAVe:WAVEform:FILEFormat, WFMInpre:NR_Pt, WFMOutpre:NR_FR?, WFMOutpre:NR_Pt?

Examples `WFMINPRE:NR_FR?` will always return `:WFMINPRE:NR_FR 1`, indicating that there is 1 data frame in the transmitted waveform record.

WFMInpre:NR_Pt

This command sets or queries the number of data points that are in the transmitted 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_FR?, WFMOutpre:NR_FR?, WFMOutpre: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 5000 specifies that 5000 data points will be transmitted.

WFMINPRE:NR_PT? might return :WFMINPRE:NR_PT 8000, indicating that there are 8000 data points in the transmitted waveform record.

WFMInpre:PT_Fmt

This command sets or queries 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 WFMOutpre: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:

Xn = XZEro + XINcr (N-PT_Off)

Ynmax = YZEro + YMUlt (ynmax - YOFf)

Ynmin = YZEro + YMUlt (ynmin - 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:

Xn = XZEro + XINcr (N-PT_Off)

Yn = YZEro + 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

This command specifies or queries the trigger point within the waveform record for the reference waveform specified by the DATa:DESTination command.

Group

Waveform Transfer

Syntax

```
WFMInpre:PT_Off <NR1>
WFMInpre:PT_Off?
```

Related Commands

DATa:DESTination, DATa:STARt, WFMOutpre:PT_Off?

Arguments

<NR1> is -(record length -1) to record length and is the position of the data point immediately following the actual trigger. <NR1> is expressed relative to DATa:STARt.

Examples

`WFMINPRE:PT_OFF 1` specifies that the trigger point is the same as the Data:Start point.

`WFMINPRE:PT_OFF?` might return `:WFMINPRE:PT_OFF 0`, indicating that the incoming waveform trigger point is the first point in the waveform record.

WFMInpre:WFId (No Query Form)

This command (no query form) accepts but ignores the argument. This command is provided only to allow a waveform extracted from the instrument to be easily imported.

Group Waveform Transfer

Syntax `WFMInpre:WFId <QString>`

Related Commands DATa:DESTination, WFMInpre:BN_Fmt, WFMInpre:ENCdg, WFMOutpre:WFId?

Arguments `<QString>` must be a valid IEEE-488.2 string (but the contents are ignored).

Examples `WFMINPRE:WFID "CH1, DC COUPLING, 2.000V/DIV, 400.0NS/DIV, 500 POINTS, SAMPLE MODE"` is a syntactically correct command.

WFMInpre:XINcr

This command sets or queries 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, WFMOutpre: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

This command sets or queries the horizontal units of the incoming waveform.

Group Waveform Transfer

Syntax

```
WFMInpre:XUNit <QString>
WFMInpre:XUNit?
```

Related Commands WFMOutpre: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

This command sets or queries the sub-sample time between the trigger sample (designated by PT_OFF) and the occurrence of the actual trigger on the incoming waveform. This value is used to compute TTOFF for the incoming waveform, and is expressed in terms of WFMInpre:XUNit.

Group Waveform Transfer

Syntax

```
WFMInpre:XZEro <NR3>
WFMInpre:XZEro?
```

Related Commands WFMInpre:PT_Off, WFMInpre:XINcr, WFMInpre:XUNit, WFMOutpre:XZEro?

Arguments <NR3> argument is a floating point value that ranges from –WFMInpre:XINcr to 0.

Examples WFMINPRE:XZERO 5.7E-6 specifies that the trigger actually occurred 5.7 µs before the sample designated by WFMInpre:PT_Off.

WFMINPRE:XZERO? might return :WFMINPRE:XZEro 7.5000E-6, indicating that the trigger occurs 7.5 µs before the sample designated by WFMInpre:PT_Off.

WFMInpre:YMUlt

This command sets or queries the vertical scale factor (in units/digitizing level) for the reference waveform, specified by DATa:DESTination, upon a CURVe command.

Group Waveform Transfer

Syntax
```
WFMInpre:YMUlt <NR3>
WFMInpre:YMUlt?
```

Related Commands DATa:DESTination, WFMInpre:BYT_Nr, WFMInpre:YUNit

Arguments <NR3> is the vertical scale factor per digitizing level of the incoming waveform points.

Examples 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).

WFMINPRE:YMULT? might return :WFMINPRE:YMULT 40.0000E-3, indicating that the vertical scale is 40 mV/digitizing level (1 V/div).

WFMInpre:YOFf

This command sets or queries the vertical position of the incoming waveform in digitizing levels. Variations in this number are analogous to changing the vertical position of the waveform. For those formats in which WFMInpre:BYT_Nr is important (all non-floating point formats), this command must take the location of the binary point implied by BYT_NR into consideration.

Group Waveform Transfer

Syntax
```
WFMInpre:YOFf <NR3>
WFMInpre:YOFf?
```

Related Commands WFMInpre:BYT_Nr, WFMInpre:YMUlt, WFMOutpre:YOFf?

Arguments <NR3> is the vertical offset in digitizing levels.

Examples WFMINPRE:YOFF 50 specifies that the zero reference point for the incoming waveform is 50 digitizing levels (2 divisions) above the center of the data range.

WFMINPRE:YOFF? might return :WFMINPRE:YOFF 25, indicating the vertical position of the incoming waveform in digitizing levels.

WFMInpre:YUNit

This command sets or queries the vertical units of the incoming waveform.

Group Waveform Transfer

Syntax
```
WFMInpre:YUNit <QString>
WFMInpre:YUNit?
```

Related Commands WFMOutpre:YUNit?

Arguments <QString> contains a maximum of three alpha characters that represent the vertical unit of measure for the incoming waveform.

Examples WFMINPRE:YUNIT "PA" specifies that the vertical units for the incoming waveform are Pascal.

WFMINPRE:YUNIT? might return :WFMINPRE:YUNIT "s", indicating the vertical units for the incoming waveform are seconds.

WFMInpre:YZEro

This command sets or queries the 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.

Group Waveform Transfer

Syntax
```
WFMInpre:YZEro <NR3>
WFMInpre:YZEro?
```

Related Commands WFMInpre:YUNit, WFMOutpre:YZEro?

Arguments <NR3> is the offset in YUNits.

Examples 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).

WFMINPRE:YZERO? might return :WFMINPRE:YZEro 7.5000E-6, indicating that the zero reference for the incoming waveform is 7.5 µV below the center of the data range (given that WFMInpre:YUNit is set to V).

WFMOutpre? (Query Only)

This query-only command queries the waveform formatting data for the waveform specified by the DATa:SOUrce command. The preamble components are considered to be of two types; formatting and interpretation. The formatting components are: ENCdg, BN_Fmt, BYT_Or, BYT_Nr, BIT_Nr. The interpretation components are derived from the DATa:SOUrce specified waveform.

Group Waveform Transfer

Syntax

```
WFMOutpre?
```

Examples WFMOUTPRE? might return the waveform formatting data as:

```
:WFMOUTPRE:BYT_NR 2;BIT_NR 16;ENCDG BINARY;BN_FMT RI;BYT_OR
MSB;WFID "Ch1, DC coupling, 200.0mV/div, 10.00us/div,
1250 points, Sample mode";NR_PT 1000;PT_FMT Y;XUNIT
"s";XINCR 80.0000E-9;XZERO 0.0000;PT_OFF 625;YUNIT "V";YMULT
31.2500E-6;YOFF 0.0000;YZERO 0.0000;NR_FR 3.
```

WFMOutpre:BIT_Nr

This command sets and queries the number of bits per waveform point that outgoing waveforms contain, as specified by the DATa:SOUrce command. Note that values will be constrained according to the underlying waveform data. This specification is only meaningful when WFMOutpre:ENCdg is set to BIN and WFMOutpre:BN_Fmt is set to either RI or RP.

Group Waveform Transfer

Syntax

```
WFMOutpre:BIT_Nr <NR1>
WFMOutpre:BIT_Nr?
```

Related Commands DATa:SOUrce, WFMOutpre:BN_Fmt, WFMOutpre:ENCdg

Arguments <NR1> number of bits per data point can be 8, 16, 32 or 64.

Examples

`WFMOUTPRE:BIT_NR 16` sets the number of bits per waveform point to 16 for incoming RI and RP binary format data.

`WFMOUTPRE:BIT_NR?` might return `:WFMOUTPRE:BIT_NR 8`, indicating that outgoing RI or RP binary format data uses 8 bits per waveform point.

WFMOutpre:BN_Fmt

This command sets or queries the format of binary data for outgoing waveforms specified by the DATa:SOUrce command.

Group

Waveform Transfer

Syntax

```
WFMOutpre:BN_Fmt {RI|RP|FP}
WFMOutpre:BN_Fmt?
```

Related Commands

DATa:SOUrce

Arguments

`RI` specifies signed integer data point representation.

`RP` specifies positive integer data point representation.

`FP` specifies single-precision binary floating point data point representation.

Examples

`WFMOUTPRE:BN_FMT FP` specifies that outgoing waveform data will be in single-precision binary floating point 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

This command sets or queries the binary field data width for the waveform specified by the DATa:SOUrce command. Note that values will be constrained according to the underlying waveform data. This specification is only meaningful when WFMOutpre:ENCdg is set to BIN, and WFMOutpre:BN_Fmt is set to either RI or RP.

Group

Waveform Transfer

Syntax

```
WFMOutpre:BYT_Nr <NR1>
WFMOutpre:BYT_Nr?
```

Related Commands DATa:SOUrce, WFMOutpre:BN_Fmt, WFMOutpre:ENCdg

Arguments <NR1> is the number of bytes per data point and can be 1, 2, 4 or 8. A value of 1 or 2 bytes per waveform point indicates channel data; 4 bytes per waveform point indicate math data; 8 bytes per waveform point indicate pixel map (DPO) data.

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

This command sets or queries which byte of binary waveform data is transmitted first, during a waveform data transfer, when data points require more than one byte. This specification only has meaning when WFMOutpre:ENCdg is set to BIN.

Group Waveform Transfer

Syntax
```
WFMOutpre:BYT_Or {LSB|MSB}
WFMOutpre:BYT_Or?
```

Related Commands WFMOutpre:ENCdg

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:ENCdg

This command 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, WFMOutpre:BYT_Or, WFMOutpre:BIT_Nr, WFMOutpre:BN_Fmt

Arguments

`ASCii` specifies that the outgoing data is to be in `ASCII` format. Waveforms internally stored as integers will be sent as <NR1> numbers, while those stored as floating point will be sent as <NR3> 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 WFMOutpre:BYT_Or.

Examples

`WFMOUTPRE:ENCDG ASCii` specifies that the outgoing waveform data will be sent in ASCII format.

`WFMOUTPRE:ENCDG?` might return `:WFMOUTPRE:ENCDG BINary`, indicating that outgoing waveform data will be sent in binary format.

WFMOutpre:NR_FR? (Query Only)

This query-only command returns the number of frames for the DATa:SOUrce waveform transmitted in response to a CURVe? query.

Group

Waveform Transfer

Syntax

```
WFMOutpre:NR_FR?
```

Related Commands

CURVe, DATa, DATa:STARt, DATa:STOP, SAVe:WAVEform, SAVe:WAVEform:FILEFormat, WFMInpre:NR_FR?, WFMInpre:NR_Pt, WFMOutpre:NR_Pt?

Examples

`WFMOUTPRE:NR_FR?` might return `:WFMOUTPRE:NR_FR 500`, indicating that there are 500 data frames to be sent.

WFMOutpre:NR_Pt? (Query Only)

This query-only command returns the number of points for the DATa:SOUrce waveform that will be transmitted in response to a CURVe? query.

Group Waveform Transfer

Syntax `WFMOutpre:NR_Pt?`

Related Commands CURVe, DATa, DATa:STARt, DATa:STOP, SAVe:WAVEform, SAVe:WAVEform:FILEFormat, WFMInpre:NR_FR?, WFMInpre:NR_Pt, WFMOutpre:NR_FR?

Examples `WFMOUTPRE:NR_PT?` might return `:WFMOUTPRE:NR_PT 5000`, indicating that there are 5000 data points to be sent.

WFMOutpre:PT_Fmt? (Query Only)

This query-only command returns the point format for the waveform specified by the DATa:SOUrce command. The format specifies a set of equations describing how the scale factors in the preamble are used to give meaning to the CURVe data points.

An error is reported if the DATa:SOUrce waveform does not exist.

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)

This query-only command returns the trigger point relative to DATa:STARt for the waveform specified by the DATa:SOUrce command.

NOTE. *This returned value is the point immediately following the actual trigger.*

Group Waveform Transfer

Syntax `WFMOutpre:PT_Off?`

Related Commands DATa:SOUrce, DATa:STARt, WFMOutpre:XZEro?

Examples `WFMOUTPRE:PT_OFF?` might return `:WFMOUTPRE:PT_OFF 251` specifying that the trigger actually occurred between points 250 and 251.

WFMOutpre:PT_ORder? (Query Only)

This query-only command specifies whether the source waveform is Fast Acquisition. A Fast Acquisition waveform is stored as a 252 (vertical) by 1000 (horizontal) point bitmap. Each point represents display intensity for that screen location. Only CURVe? query functions are allowed on Fast Acquisition waveforms.

When the WFMOutpre:PT_OR query returns Column, this indicates that the source is a Fast Acquisition waveform (and that each of 1000 possible horizontal columns being transmitted contains 252 vertical points). When the WFMOutpre:PT_OR? query returns Linear, this indicates that the source is not a Fast Acquisition waveform (and that each horizontal column being sent contains only one vertical point). Note that waveform points are transmitted in the following order: top to bottom, then left to right.

Group Waveform Transfer

Syntax `WFMOutpre:PT_ORder?`

Related Commands DATa:SOUrce

Examples `WFMOUTPRE:PT_ORDER?` might return `:WFMOUTPRE:PT_OR COL` specifying that the waveform designated by the DATa:SOUrce waveform is a Fast Acquisition waveform.

`WFMOUTPRE:PT_ORDER?` might return `:WFMOUTPRE:PT_OR LINEAR`, indicating that the source waveform is a non-Fast Acquisition waveform.

WFMOutpre:WFId? (Query Only)

This query-only command returns a string describing several aspects of the acquisition parameters for the waveform specified by the DATa:SOUrce command.

An error is reported if the DATa:SOUrce waveform does not exist.

Group Waveform Transfer

Syntax `WFMOutpre:WFId?`

Related Commands DATa:SOUrce

Returns `<QString>` contains the following comma-separated fields documented in the following tables:

Table 2-51: Waveform Suffixes

Field	Description	Examples
Source	The source identification string as it appears in the front panel scale factor readouts.	“CH1–4” “Math1–3 “Ref1–4"
Coupling	A string describing the vertical coupling of the waveform (the Source1 waveform in the case of Dual Waveform Math).	“AC coupling” “DC coupling” “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” “20.00 dB/div” “45.00 deg/div” “785.4 mrad/div” “500.0 µVs/div” “10.00 kV/s/div” “200.0 mV/div” “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” “10.00 kHz/div” “50.00 c/div”
Record Length	A string containing the number of waveform points available in the entire record. The numeric portion is given as an integer.	“500 points” “500000 points”
Acquisition Mode	A string describing the mode used to acquire the waveform.	“Sample mode” “Pk Detect mode” “Hi Res mode” “Envelope mode” “Average mode”

Examples `WFMOUTPRE:WFID?` might return `:WFMOUTPRE:WFID "Ch1, DC coupling,100.0mVolts/div,500.0µs/div,500 points, Hi Res mode"`.

WFMOutpre:XINcr? (Query Only)

This query-only command 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.

An error is reported if the DATa:SOUrce waveform does not exist.

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 (500 µs/div).

WFMOutpre:XUNit? (Query Only)

This query-only command returns the horizontal units for the waveform specified by the DATa:SOUrce command.

An error is reported if the DATa:SOUrce waveform does not exist.

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)

This query-only command returns the sub-sample time between the trigger sample (designated by PT_OFF) and the occurrence of the actual trigger for the waveform specified by the DATa:SOUrce command. This value is in units of WFMOutpre:XUNit.

An error is reported if the DATa:SOUrce waveform does not exist.

NOTE. *During steady state operation, when all control changes have settled and triggers are arriving on a regular basis, this is the only part of the preamble that changes on each acquisition.*

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 actually occurred 5.63 ns before the trigger sample.

WFMOutpre:YMUlt? (Query Only)

This query-only command returns the vertical scale factor per digitizing level in units specified by WFMOutpre:YUNit for the waveform specified by the DATa:SOUrce command. For those formats in which WFMOutpre:BYT_Nr is important (all non-floating point formats), WFMOutpre:YMUlt? must take the location of the binary point implied by BYT_NR into consideration.

An error is reported if the DATa:SOUrce waveform does not exist.

Group Waveform Transfer

Syntax `WFMOutpre:YMUlt?`

Related Commands DATa:SOUrce

Examples `WFMOUTPRE:YMULT?` might return `:WFMOUTPRE:YMULT 4.0000E-3`, indicating that the vertical scale for the corresponding waveform is 100 mV/div.

WFMOutpre:YOFf? (Query Only)

This query-only command returns the vertical offset in digitized levels for the waveform specified by the DATa:SOUrce command. For those formats in which BYT_NR is important (all non-floating point formats), this command must take the location of the binary point implied by WFMOutpre:BYT_Nr into consideration.

An error is reported if the DATa:SOUrce waveform does not exist.

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.

WFMOutpre:YUNit? (Query Only)

This query-only command returns the vertical units for the waveform specified by the DATa:SOUrce command.

An error is reported if the DATa:SOUrce waveform does not exist.

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)

This query-only command returns the vertical offset in units specified by WFMOutpre:YUNit? for the waveform specified by the DATa:SOUrce command.

An error is reported if the DATa:SOUrce waveform does not exist.

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

This command resets the zoom transforms to default values for all traces or live traces. The ZOOm query returns the current vertical and horizontal positioning and scaling of the display.

Group Zoom

Syntax
```
ZOOm {RESET|RESETLive}
ZOOm?
```

Arguments RESET resets the zoom transforms to default values for all traces.

RESETLive resets the zoom transforms to default values for live traces.

Examples ZOOM RESET resets the zoom transforms to default values for all traces.

ZOOM? might return :ZOOM:MODE 0;GRATICULE:SIZE 80;SPLIT EIGHTYTWENTY:ZOOM:SCROLL:DIRECTION STOP; LOCK 0;SPEED 1:ZOOM:ZOOM1:STATE 1; SCROLLLOCK 1;CH1:HORIZONTAL:POSITION 50.0000; SCALE 5:ZOOM:ZOOM1:CH1:VERTICAL:POSTION0.0000; SCALE 1.0000:ZOOM:ZOOM1:CH2:HORIZONTAL:POSITION 50.0000;SCALE 5:ZOOM:ZOOM1:CH2:VERTICAL:POSITION 0.0000; SCALE 1.0000:ZOOM:ZOOM1:CH3:HORIZONTAL:POSITION 50.0000; SCALE 5:ZOOM:ZOOM1:CH3:VERTICAL:POSITION 0.0000; SCALE 1.0000:ZOOM:ZOOM1:CH4:HORIZONTAL:POSITION 50.0000; SCALE 5:ZOOM:ZOOM1:CH4:VERTICAL:POSITION 0.0000;SCALE 1.0000:ZOOM:ZOOM1:MATH1:HORIZONTAL:POSITION 50.0000; SCALE 5:ZOOM:ZOOM1:MATH1:VERTICAL:POSITION 0.0000;SCALE 1.0000:ZOOM:ZOOM1:MATH2:HORIZONTAL:POSITION 50.0000; SCALE 5:ZOOM:ZOOM1:MATH2:VERTICAL:POSITION 0.0000;...

ZOOm:GRAticule:SIZE

This command sets or queries the size in percent, of the Zoom (lower) graticule.

Group Zoom

Syntax

```
ZOOm:GRAticule:SIZE {50|80|100}
ZOOm:GRAticule:SIZE?
```

Arguments

50 sets the size of the Zoom graticule to 50%.

80 set the size of the Zoom graticule to 80%.

100 sets the size of the Zoom graticule to 100%.

Examples

ZOOM:GRATICULE:SIZE 100 sets the zoom graticule size to 100%.

ZOOM:GRATICULE:SIZE? might return :ZOOM:GRATICULE:SIZE 100, indicating that the zoom graticule size is set to 100%.

ZOOm:HORizontal:POSition

This command sets or queries the horizontal position around which the zoom waveform is displayed. It is freely movable within the confines of the acquired waveform (0% to 100%) and measured from left to right of the acquired waveform.

Group Zoom

Syntax

```
ZOOm:HORizontal:POSition <NR3>
ZOOm:HORizontal:POSition?
```

Arguments

<NR3> is a value from 0 to 100.00 and is the percent of the waveform that is to the left of screen center, when the zoom factor is 1× or greater.

Examples

ZOOM:HORIZONTAL:POSITION 50 sets the horizontal position to 50.

ZOOM:HORIZONTAL:POSITION? might return :ZOOM:HORIZONTAL:POSITION 23.90000, indicating that the horizontal position is set to 23.9.

ZOOm:HORizontal:SCAle

This command sets or queries the zoom horizontal scale factor around which the zoom waveform is displayed.

Group Zoom

Syntax

```
ZOOm:HORizontal:SCAle <NR3>
ZOOm:HORizontal:SCAle?
```

Arguments

<NR3> is the amount of expansion in the horizontal direction in 1-2-5 increments.

Examples

ZOOM:HORIZONTAL:SCALE 5 sets the horizontal scale to 5.

ZOOM:HORIZONTAL:SCALE? might return :ZOOM:HORIZONTAL:SCALE 10, indicating that the horizontal scale is set to 10.

ZOOm:MATH<x>:HORizontal:POSition

This command sets or queries the horizontal position of the specified math waveform.

Group

Zoom

Syntax

```
ZOOm:MATH<x>:HORizontal:POSition <NR3>
ZOOm:MATH<x>:HORizontal:POSition?
```

Arguments

<NR3> is a value from 0 to 100.00 and is the percent of the waveform that is to the left of screen center, when the zoom factor is 1× or greater.

Examples

ZOOM:MATH1:HORIZONTAL:POSITION 50 sets the horizontal position of the specified Math 1 waveform to 50.

ZOOM:MATH1:HORIZONTAL:POSITION? might return :ZOOM:MATH1:HORIZONTAL:POSITION 23.9000, indicating that the horizontal position of the specified Math 1 waveform is set to 23.9.

ZOOm:MATH<x>:HORizontal:SCAle

This command sets or queries the zoom horizontal scale factor of the specified math waveform.

Group

Zoom

Syntax

```
ZOOm:MATH<x>:HORizontal:SCAle <NR3>
ZOOm:MATH<x>:HORizontal:SCAle?
```

Arguments <NR3> is the amount of expansion in the horizontal direction in 1-2-5 increments.

Examples ZOOM:MATH1:HORIZONTAL:SCALE 10 sets the horizontal scale factor of the specified Math 1 waveform to 10.

ZOOM:MATH1:HORIZONTAL:SCALE? might return :ZOOM:MATH1:HORIZONTAL:SCALE 5, indicating that the horizontal scale factor of the specified Math 1 waveform to 5.

ZOOm:MATH<x>:VERTical:POSition

This command sets or queries the vertical position of the specified math waveform.

Group Zoom

Syntax
```
ZOOm:MATH<x>:VERTical:POSition <NR3>
ZOOm:MATH<x>:VERTical:POSition?
```

Arguments <NR3> is the vertical position, expressed in divisions.

Examples ZOOM:MATH1:VERTICAL:POSITION 5 sets the vertical l position of the specified Math 1 waveform to 5.

ZOOM:MATH1:VERTICAL:POSITION? might return :ZOOM:MATH1:VERTICAL:POSITION 340.0000E-3, indicating that the vertical position of the specified Math 1 waveform is set to 0.34.

ZOOm:MATH<x>:VERTical:SCAle

This command sets or queries the zoom vertical scale of the specified math waveform.

Group Zoom

Syntax
```
ZOOm:MATH<x>:VERTical:SCAle <NR3>
ZOOm:MATH<x>:VERTical:SCAle?
```

Arguments <NR3> sets the horizontal scale factor of the specified Math 1 waveform to 10.

Examples `ZOOM:MATH1:VERTICAL:SCALE 5` sets the vertical scale factor of the specified Math 1 waveform to 5.

`ZOOM:MATH1:VERTICAL:SCALE?` might return `:ZOOM:MATH1:VERTICAL:SCALE 2.0000`, indicating that the vertical scale factor of the specified Math 1 waveform is set to 2.

ZOOm:MODe

This command turns Zoom mode on or off. The Zoom 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 {ON|OFF|<NR1>}
ZOOm:MODe
```

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:REF<x>:HORizontal:POSition

This command sets or queries the horizontal position of the specified reference waveform.

Group Zoom

Syntax
```
ZOOm:REF<x>:HORizontal:POSition <NR3>
ZOOm:REF<x>:HORizontal:POSition?
```

Arguments `<NR3>` is a value from 0 to 100.00 and is the percent of the waveform that is to the left of screen center, when the zoom factor is 1× or greater.

Examples `ZOOM:REF1:HORIZONTAL:POSITION 50` sets the horizontal position of the specified Reference 1 waveform to 50.

`ZOOM:REF1:HORIZONTAL:POSITION?` might return `:ZOOM:REF1:HORIZONTAL:POSITION 23.9000`, indicating that the horizontal position of the specified Reference 1 waveform is set to 23.9.

ZOOm:REF<x>:HORizontal:SCAle

This command sets or queries the zoom horizontal scale factor of the specified reference waveform.

Group Zoom

Syntax
```
ZOOm:REF<x>:HORizontal:SCAle <NR3>
ZOOm:REF<x>:HORizontal:SCAle?
```

Arguments <NR3> is the amount of expansion in the horizontal direction in 1-2-5 increments.

Examples `ZOOM:REF1:HORIZONTAL:SCALE 10` sets the horizontal scale factor of the specified Reference 1 waveform to 10.

`ZOOM:REF1:HORIZONTAL:SCALE?` might return `:ZOOM:REF1:HORIZONTAL:SCALE 5`, indicating that the horizontal scale factor of the specified Reference 1 waveform to 5.

ZOOm:REF<x>:VERTical:POSition

This command sets or queries the vertical position of the specified reference waveform.

Group Zoom

Syntax
```
ZOOm:REF<x>:VERTical:POSition <NR3>
ZOOm:REF<x>:VERTical:POSition?
```

Arguments <NR3> is the vertical position, expressed in divisions.

Examples `ZOOM:REF1:VERTICAL:POSITION 5` sets the vertical l position of the specified Reference 1 waveform to 5.

ZOOM:REF1:VERTICAL:POSITION? might return :ZOOM:REF1:VERTICAL:POSITION 340.0000E-3, indicating that the vertical position of the specified Reference 1 waveform is set to 0.34.

ZOOm:REF<x>:VERTical:SCAle

This command sets or queries the zoom vertical scale of the specified reference waveform.

Group Zoom

Syntax

```
ZOOm:REF<x>:VERTical:SCAle <NR3>
ZOOm:REF<x>:VERTical:SCAle?
```

Arguments <NR3> sets the horizontal scale factor of the specified Reference 1 waveform to 10.

Examples ZOOM:REF1:VERTICAL:SCALE 5 sets the vertical scale factor of the specified Reference 1 waveform to 5.

ZOOM:REF1:VERTICAL:SCALE? might return :ZOOM:REF1:VERTICAL:SCALE 2.0000, indicating that the vertical scale factor of the specified Reference 1 waveform is set to 2.

ZOOm:SCROLL:DIREction

This command sets or queries the direction for automatic scrolling of zoomed waveforms.

Group Zoom

Syntax

```
ZOOm:SCROLL:DIREction {FWD|FFWD|REV| FREV|STOP}
ZOOm:SCROLL:DIREction?
```

Arguments FWD starts AutoScroll. The Zoom Box moves from left to right.

FFWD starts AutoScroll. Increases the scrolling speed in the forward direction.

REV starts AutoScroll. The Zoom Box moves from right to left.

FREV starts AutoScroll. Increases the scrolling speed in the reverse direction.

STOP halts AutoScroll.

Examples `ZOOM:SCROLL:DIRECTION FWD` starts AutoScroll. The Zoom Box moves from left to right.

`ZOOM:SCROLL:DIRECTION?` might return `:ZOOM:SCROLL:DIRECTION REVERSE`, indicating that Zoom Box is moving from right to left.

ZOOm:SCROLL:LOCk

This command sets or queries the state of Scroll Lock, which "locks" zoomed waveforms under common control.

Group Zoom

Syntax
```
ZOOm:SCROLL:LOCk {ON|OFF|<NR1>}
ZOOm:SCROLL:LOCk?
```

Arguments `ON` enables Scroll Lock.

`OFF` disables Scroll Lock.

`<NR1>` = 0 disables Scroll Lock for all zoom windows; any other value enables Scroll Lock for all zoom windows.

Examples `ZOOM:SCROLL:LOCK ON` enables Scroll Lock for zoomed waveforms identified as locked.

`ZOOM:SCROLL:LOCK?` might return `:ZOOM:SCROLL:LOCK OFF`, indicating that the scroll lock function is disabled.

ZOOm:SCROLL:SPEED

This command sets or queries the speed of automatic scrolling.

Group Zoom

Syntax
```
ZOOm:SCROLL:SPEED <NR1>
ZOOm:SCROLL:SPEED?
```

Arguments `<NR1>` is a value from 1 to 10.

Examples `ZOOM:SCROLL:SPEED 2` will scroll zoomed waveform(s) 2 divisions per update.

`ZOOM:SCROLL:SPEED?` might return `:ZOOM:SCROLL:SPEED 1`, indicating that the scroll speed is set to move zoomed waveform(s) 1 division per update.

ZOOm:STATE

This command sets or queries the specified zoom on or off.

Group Zoom

Syntax
```
ZOOm:STATE {ON|OFF|<NR1>}
ZOOm:STATE?
```

Arguments `ON` turns zoom on or off.

`OFF` turns zoom on or off.

`<NR1>` = 0 disables the zoom; any other value enables the zoom.

Examples `ZOOM:STATE ON` turns zoom on.

`ZOOM:STATE?` might return `:ZOOM:STATE 1`, indicating that zoom is on.

ZOOm:VERTical:POSition

This command sets or queries the vertical position around which the zoom waveform is displayed. It is freely movable within the confines of the acquired waveform (0% to 100%) and measured from left to right of the acquired waveform.

Group Zoom

Syntax
```
ZOOm:VERTical:POSition <NR3>
ZOOm:VERTical:POSition?
```

Arguments `NR3` is a value from 0 to 100.00 and is the percent of the waveform that is to the left of screen center, when the zoom factor is 1× or greater.

Examples `ZOOM:VERTICAL:POSITION 50` sets the vertical position to 50.

`ZOOM:VERTICAL:POSITION?` might return `:ZOOM:VERTICAL:POSITION 23.90000`, indicating that the vertical position is set to 23.9.

ZOOm:VERTical:SCAle

This command sets or queries the zoom vertical scale factor around which the zoom waveform is displayed.

Group Zoom

Syntax

```
ZOOm:VERTical:SCAle <NR3>
ZOOm:VERTical:SCAle?
```

Arguments <NR3> is the amount of expansion in the vertical direction in 1-2-5 increments.

Examples ZOOM:VERTICAL:SCALE 5 sets the vertical scale to 5.

ZOOM:VERTICAL:SCALE? might return :ZOOM:VERTICAL:SCALE 10, indicating that the vertical scale is set to 10.

ZOOm:ZOOM<x>

This command resets the zoom transforms to default values for all traces of the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window. The ZOOm:REF<x>:VERTical:SCAle? query returns the current vertical and horizontal positioning and scaling of the display.

Group Zoom

Syntax

```
ZOOm:ZOOM<x> {RESET|RESETLive}
ZOOm:ZOOM<x>?
```

Arguments RESET resets the zoom transforms to default values for all traces of the specified zoom.

RESETLive resets the zoom transforms to default values for live traces of the specified zoom.

Examples ZOOM:ZOOM1 RESET resets the zoom transforms for all traces of Zoom1 to their default values

ZOOM:ZOOM1? might return :ZOOM:ZOOM1:STATE 1;SCROLLLOCK 1; CH1:HORIZONTAL POSITION 50.0000;SCALE 5; :ZOOM:ZOOM1:CH1:VERTICAL:POSITION 0.0000; SCALE 1.0000; :ZOOM:ZOOM1:CH2:HORIZONTAL:POSTION 50.0000;

SCALE 5; :ZOOM:ZOOM1:CH2:VERTICAL:POSITION 0.000; SCALE 1.0000; :ZOOM:ZOOM1:CH3:HORIZONTAL:POSITION50.000; SCALE 5;:ZOOM:ZOOM1:CH3:VERTICAL..., indicating the Zoom1 transforms for all traces.

ZOOm:ZOOM<x>:DCHAN:DISplay

This command sets or queries the display of the digital waveforms for the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window. The digital waveforms affected vary from CH1, CH2, CH3, CH4, Math1, Math2, Math3, Math4, and D0 through D15. When zoom is set on any one of the waveforms from D0 through D15, it would apply to all of the waveforms (D0 through D15).

Group Zoom

Syntax

```
ZOOm:ZOOM<x>:DCHAN:DISplay {<NR1>|OFF|ON}
ZOOm:ZOOM<x>:DCHAN:DISplay?
```

Arguments

<NR1> = 0 disables the specified zoom; any other value enables the specified zoom.

OFF disables the specified zoom.

ON displays the specified zoom.

Examples

ZOOM:ZOOM1:DCHAN:DISPLAY ON displays the digital waveforms for the specified Zoom1.

ZOOM:ZOOM1:DCHAN:DISPLAY? might return :ZOOM:ZOOM1:DCHAN:DISPLAY 0, indicating that the display of the digital waveforms for the specified Zoom1 is disabled.

ZOOm:ZOOM<x>:SCROLLLock

This command sets or queries Scroll Lock for the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window.

Group Zoom

Syntax

```
ZOOm:ZOOM<x>:SCROLLLock {ON|OFF|<NR1>}
ZOOm:ZOOM<x>:SCROLLLock?
```

Arguments ON locks waveforms for the specified zoom window <x>.

OFF unlocks waveforms for the specified zoom window <x>.

<NR1> = 0 unlocks waveforms for the specified zoom window <x>; any other value locks waveforms for the specified zoom window <x>.

Examples ZOOM:ZOOM1:SCROLLLOCK ON turns Zoom1 scroll lock on.

ZOOM:ZOOM1:SCROLLLOCK? returns either ON|1 or OFF|0 for Zoom1.

ZOOm:ZOOM<x>:STATE

This command sets or queries the specified zoom on or off, where x is an integer from 1 to 4 representing the desired zoom window.

Default is Zoom1 on, Zoom2 - 4 off.

Group Zoom

Syntax
```
ZOOm:ZOOM<x>:STATE {ON|OFF|<NR1>}
ZOOm:ZOOM<x>:STATE?
```

Arguments ON turns Zoom 1-4 on.

OFF turns Zoom 1-4 off.

<NR1> = 0 disables the specified zoom; any other value enables the specified zoom.

Examples ZOOM:ZOOM1:STATE ON turns Zoom1 on.

ZOOM:ZOOM2:STATE? might return :ZOOM:ZOOM2:STATE 1, indicating that Zoom2 is on.

ZOOm:ZOOM<x>:<wfm>:DISplay

This command sets or queries the display of the specified waveform for the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window. The waveform affected is determine by <wfm> which can vary from CH1, CH2, CH3, CH4, Math1, Math2, Math3, Math4, Ref1, Ref2, Ref3, or Ref4.

Group Zoom

Syntax

```
ZOOm:ZOOM<x>:<wfm>:DISplay {<NR1>|OFF|ON}
ZOOm:ZOOM<x>:<wfm>:DISplay?
```

Arguments

<NR1> A zero disables the specified zoom; any other value enables the specified zoom.

OFF disables the specified zoom.

ON displays the specified zoom.

Examples

ZOOM:ZOOM1:CH1:DISPLAY ON displays the specified zoom waveform.

ZOOM:ZOOM1:CH1:DISPLAY? might return :ZOOM:ZOOM1:CH1:DISPLAY 0, indicating the channel 1 zoom is disabled.

ZOOm:ZOOM<x>:<wfm>:HORizontal:POSition

This command sets or queries the horizontal position of the specified waveform for the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window. The setting of the ZOOm:ZOOM<x>:SCROLLLock command determines the waveforms that are affected.

The waveform affected is determine by <wfm> which can vary from CH1, CH2, CH3, CH4, Math1, Math2, Math3, Math4, Ref1, Ref2, Ref3, or Ref4.

Group

Zoom

Syntax

```
ZOOm:ZOOM<x>:<wfm>:HORizontal:POSition <NR3>
ZOOm:ZOOM<x>:<wfm>:HORizontal:POSition?
```

Arguments

<NR3> is a value from 0 to 100.00 and is the percent of the waveform that is to the left of screen center, when the zoom factor is 1× or greater.

Examples

ZOOM:ZOOM1:CH1:HORIZONTAL:POSITION 50 sets the Zoom1 reference pointer at 50% of acquired waveform.

ZOOM:ZOOM1:CH1:HORIZONTAL:POSITION? might return :ZOOM1:CH1:HORIZONTAL:POSITION 50.0000, indicating that the Zoom1 reference pointer for Channel 1 is currently set at 50% of acquired waveform.

ZOOm:ZOOM<x>:<wfm>:HORizontal:SCAle

This command sets or queries the zoom horizontal scale factor of the specified waveform for the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window.

The waveform affected is determine by <wfm> which can vary from CH1, CH2, CH3, CH4, Math1, Math2, Math3, Math4, Ref1, Ref2, Ref3, or Ref4.

Group Zoom

Syntax

```
ZOOm:ZOOM<x>:<wfm>:HORizontal:SCAle <NR3>
ZOOm:ZOOM<x>:<wfm>:HORizontal:SCAle?
```

Arguments <NR3> is the amount of expansion in the horizontal direction in 1-2-5 increments.

Examples `ZOOM:ZOOM1:CH1:HORIZONTAL:SCALE 5` sets the horizontal scale factor of Channel 1 to 5.

`ZOOM:ZOOM2:CH2:HORIZONTAL:SCALE?` might return `:ZOOM2:CH2:HORIZONTAL:SCALE 1`, indicating that Channel 2 has a horizontal scale factor of 1.

ZOOm:ZOOM<x>:<wfm>:VERTical:POSition

This command sets or queries the vertical position of the specified waveform for the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window. The setting of the `ZOOm:ZOOM<x>:SCROLLLOCk` command determines the waveforms that are affected. The <wfm> can be a channel, math, or reference waveform.

The waveform affected is determine by <wfm> which can vary from CH1, CH2, CH3, CH4, Math1, Math2, Math3, Math4, Ref1, Ref2, Ref3, or Ref4.

Group Zoom

Syntax

```
ZOOm:ZOOM<x>:<wfm>:VERTical:POSition <NR3>
ZOOm:ZOOM<x>:<wfm>:VERTical:POSition?
```

Arguments <NR3> is the vertical position, expressed in divisions.

Examples

`ZOOM:ZOOM1:CH1:VERTICAL:POSITION 2` sets the Zoom1 vertical position to 2, which centers the zoom trace at the second division of the acquired trace.

`ZOOM:ZOOM1:CH1:VERTICAL:POSITION?` might return `:ZOOm1:CH1:VERTical:POSition 0.0000`, indicating that the Zoom1 trace for Channel 1 is centered at division 0 of the acquired trace.

ZOOm:ZOOM<x>:<wfm>:VERTical:SCAle

This command sets or queries the zoom vertical scale of the specified waveform for the specified zoom, where x is an integer from 1 to 4 representing the desired zoom window.

The waveform affected is determine by <wfm> which can vary from CH1, CH2, CH3, CH4, Math1, Math2, Math3, Math4, Ref1, Ref2, Ref3, or Ref4.

Group

Zoom

Syntax

```
ZOOm:ZOOM<x>:<wfm>:VERTical:SCAle <NR3>
ZOOm:ZOOM<x>:<wfm>:VERTical:SCAle?
```

Arguments

<NR3> is the amount of vertical expansion or compression, which operates on a 1-2-5 sequence (for example, 1, 2, 5, 10, 20, 50, 100...). Based on the value that you entered, this command uses the nearest scale factor. Setting the vertical scale to 1 indicates unity (no zoom).

Examples

`ZOOM:ZOOM2:CH2:VERTICAL:SCALE 5` sets the vertical scale of Channel 2 to 5×.

`ZOOM:ZOOM2:CH2:VERTICAL:SCALE?` might return `:ZOOM2:CH2:VERTICAL:SCALE 2.0000`, indicating that the vertical scale is 2×.

Status and events

The instrument provides a status and event reporting system for the GPIB interfaces. This system informs you of certain significant events that occur within the instrument.

The instrument 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 instrument. 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

The Standard Event Status Register (SESR) and the Status Byte Register (SBR) record certain types of events that can occur while the instrument 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 instrument 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 instrument. 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

Table 3-1: SESR Bit Functions

Bit	Function	
7 (MSB)	PON	Power On. Shows that the instrument was powered on. On completion, the diagnostic tests also set this bit.
6	URQ	User Request. Indicates that an application event has occurred. *See preceding note.
5	CME	Command Error. Shows that an error occurred while the instrument 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.

Status Byte Register (SBR). Records whether output is available in the Output Queue, whether the instrument 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.

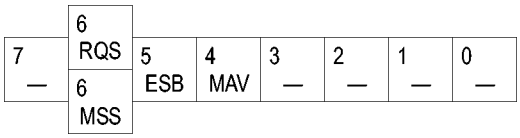


Table 3-2: SBR bit functions

Bit	Function	
7 (MSB)	———	Not used.
6	RQS	Request Service. Obtained from a serial poll. Shows that the instrument 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.
4	MAV	Message Available. Shows that output is available in the Output Queue.

Table 3-2: SBR bit functions (cont.)

Bit	Function
3	———— Not used.
2	———— Not used.
1–0	———— Not used.

Enable

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 as follows.

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

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

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	—	—	—	—

*PSC command

*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. *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 instrument 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 instrument 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, can 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 more than 32 events stack up in the Event Queue, the 32nd event is replaced by event code 350, “Queue Overflow.”

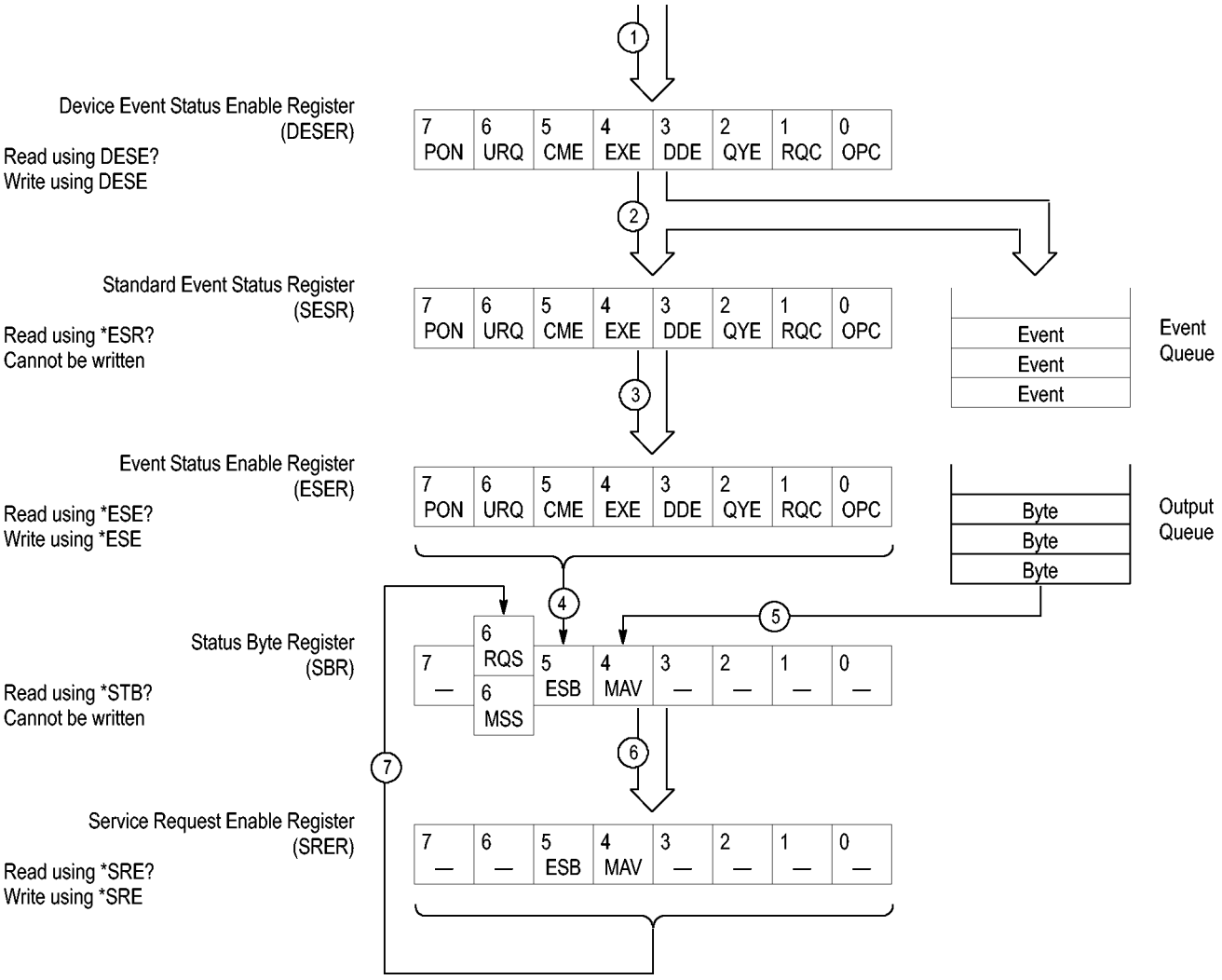
Read the Event Queue with the EVENT? query (which returns only the event number), with the EVMSG? query (which returns the event number and a text description of the event), or with the ALLEV? query (which returns all the event numbers with a description of the event). Reading an event removes it from 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.



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

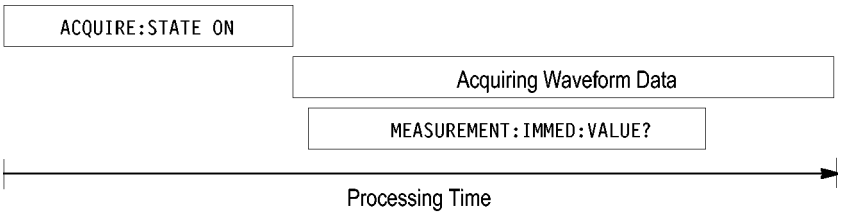
Although most GPIB commands are completed almost immediately after being received by the instrument, some commands start a process that requires more time. For example, once a single sequence acquisition command is executed, depending upon the applied signals and trigger settings, it can be a few seconds before the acquisition is complete. Rather than remain idle while the operation is in process, the instrument 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 gets processed. The instrument status and event reporting system provides ways to do this.

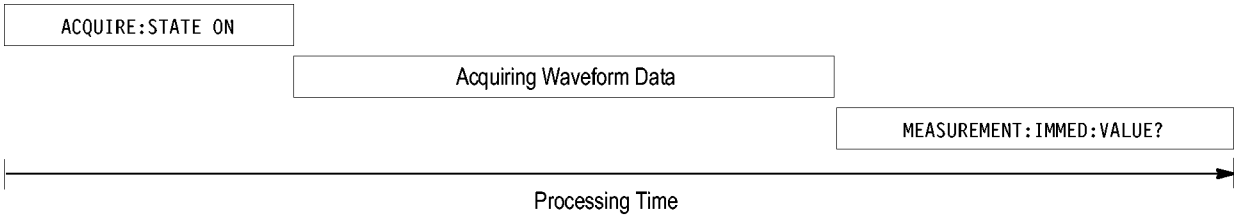
For example, a typical application might involve acquiring a single-sequence waveform and then taking a measurement on the acquired waveform. You could use the following command sequence to do this:

```
/** Set up conditional acquisition **/
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:MODE: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:IMMED:VALUE?
```

The acquisition of the waveform requires extended processing time. It might not finish before the instrument takes an amplitude measurement (see the following figure). This can result in an incorrect amplitude value.



To ensure that the instrument completes waveform acquisition before taking the measurement on the acquired data, you can synchronize the program.



You can use four commands to synchronize the operation of the instrument with your application program: *WAI, BUSY, *OPC, and *OPC?

***WAI** 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:MODE: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 instrument, but the commands will not be processed by the instrument until all OPC operations in process are complete. If the input buffer becomes full, the

controller will be unable to write more commands to the buffer. This can cause a time-out.

BUSY The BUSY? query allows you to find out whether the instrument 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:MODE: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 more bus traffic.

*OPC

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:MODE: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, a Service Request will be generated.

The same command sequence using the *OPC command for synchronization looks like this

/* Set up conditional acquisition */

ACQUIRE:STATE OFF

SELECT:CH1 ON

HORIZONTAL:MODE: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 more efficient but requires more sophisticated programming.

*OPC?

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:MODE: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

Overview. The information contained in the topic tabs above covers all the programming interface messages the instrument generates in response to commands and queries.

For most messages, a secondary message from the instrument gives more 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
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
111	Header separator error
112	Program mnemonic too long
113	Undefined header
114	Header suffix out of range
120	Numeric data error
121	Invalid character in numeric
123	Exponent too large
124	Too many digits
128	Numeric data not allowed
130	Suffix error
131	Invalid suffix
134	Suffix too long
138	Suffix not allowed
140	Character data error
141	Invalid character data
144	Character data too long
148	Character data not allowed
150	String data error
151	Invalid string data
158	String data not allowed
160	Block data error
161	Invalid block data
168	Block data not allowed
170	Command expression error
171	Invalid expression
178	Expression data not allowed
180	Macro error
181	Invalid outside macro definition

Table 3-4: Command error messages (CME Bit 5) (cont.)

Code	Message
183	Invalid inside macro definition
184	Macro parameter error
190	Communication option not present
191	Unable to open or parse arbitrary filter file

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)

Code	Message
200	Execution error
201	Invalid while in local
202	Settings lost due to RTL
210	Trigger error
211	Trigger ignored
212	Arm ignored
213	Init ignored
214	Trigger deadlock
215	Arm deadlock
219	Trigger delay not available
220	Parameter error
221	Settings conflict
222	Data out of range
223	Too much data
224	Illegal parameter value
225	Out of memory
226	Lists not same length
230	Data corrupt or stale
231	Data questionable
240	Hardware error
241	Hardware missing
244	Invalid printer selected
245	Hardcopy/Export operation timed out
250	Mass storage error
251	Missing mass storage
252	Missing media
253	Corrupt media

Table 3-5: Execution error messages (EXE Bit 4) (cont.)

Code	Message
254	Media full
255	Directory full
256	File name not found
257	File name error
258	Media protected
259	File system error
260	Execution expression error
261	Math error in expression
270	Execution macro error
271	Macro syntax error
272	Macro execution error
273	Illegal macro label
274	Execution macro parameter error
275	Macro definition too long
276	Macro recursion error
277	Macro redefinition not allowed
278	Macro header not found
280	Program error
281	Cannot create program
282	Illegal program name
283	Illegal variable name
284	Program currently running
285	Program syntax error
286	Program runtime error
2200	Measurement error, Measurement system error
2201	Measurement error, Zero period
2202	Measurement error, No period found
2203	Measurement error, No period, second waveform
2204	Measurement error, Low signal amplitude
2205	Measurement error, Low amplitude, second waveform
2206	Measurement error, Invalid gate
2207	Measurement error, Measurement overflow
2208	Measurement error, Waveform does not cross Mid Ref
2209	Measurement error, No second Mid Ref crossing
2210	Measurement error, No Mid Ref crossing, second waveform
2211	Measurement error, No backward Mid Ref Crossing
2212	Measurement error, No negative crossing

Table 3-5: Execution error messages (EXE Bit 4) (cont.)

Code	Message
2213	Measurement error, No positive crossing
2214	Measurement error, No crossing
2215	Measurement error, No crossing, second waveform
2216	Measurement error, No crossing, target waveform
2217	Measurement error, Constant waveform
2218	Measurement error, Unused
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
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
2241	Waveform requested is invalid
2243	The measurement cannot be turned on because the source for the measurement is incompatible
2244	Source waveform is not active
2248	This reference waveform cannot be activated
2249	Reference deletion error, the reference waveform is in use for math
2250	Reference error, the reference waveform file is invalid
2251	Reference error, Waveform preamble does not match actual
2252	Reference error, Source waveform is not valid
2253	Reference error, Too many points received
2254	Reference error, Too few points received
2255	Reference error, destination reference is used by a math source
2256	Reference error, cannot reposition or rescale a reference waveform that was saved in FastAcq mode
2257	Reference warning, saving a previewed waveform results in an empty reference
2258	Reference warning, saving a FastAcq waveform may result in an empty reference
2259	File too big

Table 3-5: Execution error messages (EXE Bit 4) (cont.)

Code	Message
2261	Calibration error, wait for the warm-up interval to expire before invoking SPC
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
2287	You have 4 free trials left for this application
2288	You have 3 free trials left for this application
2289	You have 2 free trials left for this application
2294	You have 1 free trial left for this application
2295	This is the last free trial for this application
2296	To guarantee accuracy the vertical amplifiers must use the same coupling and the same model probes
2297	CH1 and AUX cannot be selected as the A trigger event and the B trigger event respectively or vice versa
2298	CAL: failed to read NVram, will use uncalibrated factory defaults. The scope must be calibrated at the factory
2299	Failed to read SPC Calibration file, will use uncalibrated factory defaults until SPC is performed
2309	Video Trigger does not allow selection of default trigger holdoff
2310	CAL: invalid command
2312	Deskew not applied when FastAcq is on
2313	Probe error, wait for the warmup interval to expire before doing probe calibration
2318	Consider system bandwidth when testing at this bit rate
2319	Deskew is re-enabled
2321	Histogram warning, histogram turned off because Zoom or FastFrame was enabled
2322	Histogram warning, turning histogram on turned other features off
2330	FastFrame time stamp query requested too many time stamps
2350	When FastFrame is active, the reference waveform source must be displayed
2351	Acquire Modes (Avg|Env) incompatible with FastFrame
2352	Acquire Modes (Avg|Env) incompatible with FastFrame. Set to Sample
2360	FastAcq warning, FastAcq turned Off
2370	Zoom warning, Zoom turned Off
2371	Zoom error, Zoom has circular definition

Table 3-5: Execution error messages (EXE Bit 4) (cont.)

Code	Message
2372	Zoom error, Source zoom is not active
2400	Not enough memory available
2401	This channel cannot be activated
2402	Math/Meas/Histo have circular definition
2403	Cannot enter XYZ mode because Channel 4 is active (in use)
2404	XY waveform error, cannot activate channel while in XY mode
2405	XYZ waveform error, cannot activate channel while in XYZ mode
2410	Empty math string
2411	Syntax error in math string
2412	Semantic error in math string
2413	Math expression is too complex
2414	Spectral Function Syntax Error, can operate only on single Channel or Math source
2415	Spectral Function Semantic Error, cannot operate on static (Reference) sources
2420	Histogram cannot be performed on this type of waveform
2425	Mask counting cannot be performed on this type of wfm
2430	WfmDB cannot be built on this type of wfm
2470	Trigger error, cannot enter Random holdoff unless type is set to Edge
2471	Trigger error, cannot enter Random holdoff unless source is set to Channel 1-4
2472	Trigger error; cannot enter Random holdoff unless A->B Sequence is set to A Only
2473	Trigger T cannot be turned on when in XY display format mode
2474	Trigger slope not available, coerced to Rise
2475	Either trigger slope not available when in MultiScope AND mode
2476	Trigger type not available in MultiScope AND mode
2477	Pattern lock trigger not available when MultiScope triggering is enabled
2500	Setup error, file does not look like a setup file
2501	Setup warning, could not recall all values from external setup
2510	iCapture source selection failed; new source already in use
2511	Busses may not contain iCapture signals as members
2512	iCapture sources were reassigned to support serial bus triggering
2513	Clocked Parallel Bus clock source changed because previous source was added as member signal
2600	Mask testing complete
2601	Mask testing failed
2610	Standard Mask Autoset failed, not allowed for Math or Ref Waveforms

Table 3-5: Execution error messages (EXE Bit 4) (cont.)

Code	Message
2611	Standard Mask Autoset failed, source of selected mask type not found
2612	Turn off Optical Filtering to allow this to work
2613	Optical Filtering not allowed
2614	Autoset is not allowed during Edit User Mask
2615	Mask setup error, file does not look like a setup file
2620	Mask error, too few points received
2700	Limit Test Error: Reference and source channel have different horizontal settings
2701	Limit Test Failure: Incompatible record lengths
2702	Template creation error: Source waveform not compatible
2703	Template creation error: Target reference is being used in an active limit test
2704	Limit Test Error: Reference is not a valid template
2705	Limit Test Failure: Cannot compare a reference to itself
2706	Limit Test Error: Disk Full
2707	Limit testing failed
2708	Select a source and template to compare before turning on the limit test
2740	No SMTP Server address is entered
2741	Sending E-mail...
2742	Email on Trigger is disabled when Fast Acquisitions are enabled
2743	Email error; must have measurements active before selecting this option
2750	H-Bar cursors are not supported in xy mode
2751	V-Bar cursors are not supported in xy mode
2752	Cursor Error; the current source is invalid or inactive
2760	Mark limit reached
2761	No mark present
2762	Search copy failed
2763	Unable to recall the Marks
2764	Reference error, the reference waveform file is unsupported
2765	The Search cannot be turned on due to FastAcq or WfmDB mode
2766	Advanced Search Option not present
2767	Inputs CH1-CH4 incompatible; set state clock source to CH4
2768	Inputs incompatible; first set CH1-CH4 inputs to XXXX
2769	Search Mark sources must have matching record lengths
2770	Trigger Pulse source must be CH1-CH4 for this pulse class
2771	This Trigger Pulse class is not available for D0-D15
2772	D15-D0 is already used by A Event
2773	D15-D0 is already used by B Event

Table 3-5: Execution error messages (EXE Bit 4) (cont.)

Code	Message
2774	D15-D0 is already used by B Event Reset
2775	D15-D0 is already used by A Event qualification
2776	D15-D0 is already used by B Event qualification
2777	Busses cannot contain both live and reference signals
2778	Attempt to multiply add a signal to a bus

Device error. The following table lists the device errors that can occur during instrument operation. These errors might indicate that the instrument needs repair.

Table 3-6: Device error messages (DDE Bit 3)

Code	Message
300	Device-specific error
310	System error
311	Memory error
312	PUD memory lost
313	Calibration memory lost
314	Save/recall memory lost
315	Configuration memory lost
330	Self test failed
350	Queue overflow (does not set DDE bit)

System event. The following table lists the system event messages. These messages are generated whenever certain system conditions occur.

NOTE. *The only system event messages available for reading back with the EVENT? or ALLEV? queries are events that have a bit indicated in the following table.*

Table 3-7: System event messages

Code	Message
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)

Table 3-7: System event messages (cont.)

Code	Message
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
469	App button pressed
470	TekScope minimized
471	TekScope restored
472	Application variable changed
485	Ch1 probe button pressed
486	Ch2 probe button pressed
487	Ch3 probe button pressed
488	Ch4 probe button pressed
489	Aux probe button pressed

Execution warning. The following table lists warning messages that do not interrupt the flow of command execution. These notify you that you might get unexpected results.

Table 3-8: Execution warning messages (EXE Bit 4)

Code	Message
532	Curve data too long, Curve truncated
533	Curve error, Preamble values are inconsistent
535	Curve error, Reference data sources not allowed in streaming or next-waveform mode
536	Curve error, Acquisition mode must be set to Run for streaming or next-waveform mode
537	Curve error, Cannot stream or next-waveform math waveforms while in roll mode
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

Table 3-8: Execution warning messages (EXE Bit 4) (cont.)

Code	Message
551	FASTAcq mode is active – deactivate to use math
590	Diagnostic warning, running GPIB diagnostic over GPIB may affect talk/listen program

Internal warning. The following table shows internal errors that indicate an internal fault in the instrument.

Table 3-9: Internal warning messages

Code	Message
600	Internal warning
630	Internal warning, Input overload

External application error. The following table shows external application errors in the instrument.

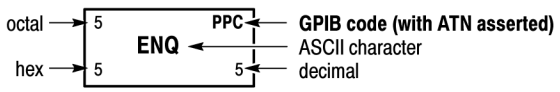
Table 3-10: External application error

Code	Message
2800	External application error

Appendix A: Character set

B7 B6 B5 BITS B4 B3 B2 B1	0 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 SYMBOLS		UPPER CASE		LOWER CASE	
0 0 0 0	0 NUL 0 0	20 DLE 10 16	40 LA0 SP 20 32	60 LA16 0 30 48	100 TA0 @ 40 64	120 TA16 P 50 80	140 SA0 ` 60 96	160 SA16 p 70 112
0 0 0 1	1 GTL SOH 1 1	21 LL0 DC1 11 17	41 LA1 ! 21 33	61 LA17 1 31 49	101 TA1 A 41 65	121 TA17 Q 51 81	141 SA1 a 61 97	161 SA17 q 71 113
0 0 1 0	2 STX 2 2	22 DC2 12 18	42 LA2 " 22 34	62 LA18 2 32 50	102 TA2 B 42 66	122 TA18 R 52 82	142 SA2 b 62 98	162 SA18 r 72 114
0 0 1 1	3 ETX 3 3	23 DC3 13 19	43 LA3 # 23 35	63 LA19 3 33 51	103 TA3 C 43 67	123 TA19 S 53 83	143 SA3 c 63 99	163 SA19 s 73 115
0 1 0 0	4 SDC EOT 4 4	24 DCL DC4 14 20	44 LA4 $ 24 36	64 LA20 4 34 52	104 TA4 D 44 68	124 TA20 T 54 84	144 SA4 d 64 100	164 SA20 t 74 116
0 1 0 1	5 PPC ENQ 5 5	25 PPU NAK 15 21	45 LA5 % 25 37	65 LA21 5 35 53	105 TA5 E 45 69	125 TA21 U 55 85	145 SA5 e 65 101	165 SA21 u 75 117
0 1 1 0	6 ACK 6 6	26 SYN 16 22	46 LA6 & 26 38	66 LA22 6 36 54	106 TA6 F 46 70	126 TA22 V 56 86	146 SA6 f 66 102	166 SA22 v 76 118
0 1 1 1	7 BEL 7 7	27 ETB 17 23	47 LA7 ' 27 39	67 LA23 7 37 55	107 TA7 G 47 71	127 TA23 W 57 87	147 SA7 g 67 103	167 SA23 w 77 119
1 0 0 0	10 GET BS 8 8	30 SPE CAN 18 24	50 LA8 (28 40	70 LA24 8 38 56	110 TA8 H 48 72	130 TA24 X 58 88	150 SA8 h 68 104	170 SA24 x 78 120
1 0 0 1	11 TCT HT 9 9	31 SPD EM 19 25	51 LA9) 29 41	71 LA25 9 39 57	111 TA9 I 49 73	131 TA25 Y 59 89	151 SA9 i 69 105	171 SA25 y 79 121
1 0 1 0	12 LF A 10	32 SUB 1A 26	52 LA10 * 2A 42	72 LA26 : 3A 58	112 TA10 J 4A 74	132 TA26 Z 5A 90	152 SA10 j 6A 106	172 SA26 z 7A 122
1 0 1 1	13 VT B 11	33 ESC 1B 27	53 LA11 + 2B 43	73 LA27 ; 3B 59	113 TA11 K 4B 75	133 TA27 [5B 91	153 SA11 k 6B 107	173 SA27 { 7B 123
1 1 0 0	14 FF C 12	34 FS 1C 28	54 LA12 , 2C 44	74 LA28 < 3C 60	114 TA12 L 4C 76	134 TA28 \ 5C 92	154 SA12 l 6C 108	174 SA28 ¦ 7C 124
1 1 0 1	15 CR D 13	35 GS 1D 29	55 LA13 – 2D 45	75 LA29 = 3D 61	115 TA13 M 4D 77	135 TA29] 5D 93	155 SA13 m 6D 109	175 SA29 } 7D 125
1 1 1 0	16 SO E 14	36 RS 1E 30	56 LA14 . 2E 46	76 LA30 > 3E 62	116 TA14 N 4E 78	136 TA30 ^ 5E 94	156 SA14 n 6E 110	176 SA30 ~ 7E 126
1 1 1 1	17 SI F 15	37 US 1F 31	57 LA15 / 2F 47	77 UNL ? 3F 63	117 TA15 O 4F 79	137 UNT _ 5F 95	157 SA15 o 6F 111	177 RUBOUT (DEL) 7F 127
	ADDRESSED COMMANDS	UNIVERSAL COMMANDS	LISTEN ADDRESSES		TALK ADDRESSES		SECONDARY ADDRESSES OR COMMANDS	

KEY



Tektronix
REF: ANSI STD X3.4-1977
IEEE STD 488.1-1987
ISO STD 646-2973

Appendix B: Reserved words

This is a list of reserved words for your instrument. Capital letters identify the required minimum spelling.

*CAL
*CLS
*DDT
*ESE
*ESR
*IDN
*LRN
*OPC
*OPT
*PSC
*PUD
*RCL
*RST
*SAV
*SDS
*SRE
*STB
*TRG
*TST
*WAI
A
ABSolute
AC
ACCEPTCOUNT
ACCEPTTIMEout
ACCept
ACCepts
ACK
ACKERRreport
ACKErrorreport
ACKMISS
ACKnum
ACQDURATION
ACQLENGTH
ACQuire
ACTIVE
ACTUal
ACTivate
ADDR10
ADDR7
ADDRANDDATA
ADDRess
ADDress
ADVANCEafter
ADVanced
ALERT
ALIGN
ALIGNCHARacter
ALIGNPRIMitive
ALIas
ALL
ALLEv
ALLFields
ALLLines
ALLMatched
ALLOcate
ALLQString
AMI
AMPlitude
ANALOgenable
ANALog
AND
ANNOTation
ANY
ANY8B10B
ANYControl
ANY_CJTPAT_BIT
ANY_CJTPAT_CHAR
AOFFSet
APPLY
APPLYtoall
APPLication
AREA
AREACOLOr
AREa
ASCIi
ASPECTratio
ATAG1
ATAG2
ATAG3
ATARXG1
ATARXG2
ATARXG3
ATATXG1
ATATXG2
ATATXG3
ATRIGger
ATTEN1000X
ATTEN100X
ATTEN10X
ATTEN125X
ATTEN12_5X
ATTEN150X
ATTEN15X
ATTEN175X
ATTEN17_5X
ATTEN1X
ATTEN1_25X
ATTEN1_5X
ATTEN1_75X
ATTEN200X
ATTEN20X
ATTEN250X
ATTEN25X
ATTEN2X
ATTEN2_5X
ATTEN300X
ATTEN30X
ATTEN350X
ATTEN35X
ATTEN3X
ATTEN3_5X
ATTEN400X
ATTEN40X
ATTEN450X
ATTEN45X
ATTEN4X
ATTEN4_5X
ATTEN500X
ATTEN50X
ATTEN550X
ATTEN55X
ATTEN5X
ATTEN5_5X
ATTEN600X
ATTEN60X
ATTEN650X
ATTEN65X
ATTEN6X
ATTEN6_5X
ATTEN700X
ATTEN70X
ATTEN750X
ATTEN75X
ATTEN7X
ATTEN7_5X
ATTEN800X
ATTEN80X
ATTEN850X
ATTEN85X
ATTEN8X
ATTEN8_5X
ATTEN900X
ATTEN90X
ATTEN950X
ATTEN95X
ATTEN9X
ATTEN9_5X
ATTempts
AUTHLogin
AUTHPassword
AUTO
AUTOAdjust
AUTOBitrate
AUTOBright
AUTOInc
AUTOReflevel
AUTOSCale
AUTOSet
AUTOZero
AUX
AUXIn
AUXLevel
AUXiliary
AUXout
AVErage
AVGmean
B
B0
B1
B10
B11
B12
B13
B14
B15
B16
B2
B3
B3ZS
B4
B5
B6
B6ZS

B7
B8
B8ZS
B9
BACKLight
BACKWards
BADCHARS
BANdwidth
BASe
BAUd
BCR
BDIFFBP
BEEP
BELl
BETWeen
BIAS
BILevelcustom
BINary
BIT
BITCOUNT
BITDURATION
BITOrder
BITRate
BITSTUFFing
BITString
BIT_Nr
BLACKANDWhite
BLACKMANHarris
BLANK
BLUe
BM
BMP
BN_Fmt
BOFFSet
BOTh
BOX
BOXPcnt
BTA
BTRIGger
BULK
BURst
BUS
BUSY
BY
BYT_Nr
BYT_Or
Block
C
CALProbe
CALibrate
CAN
CANH
CANL
CAPS
CAREa
CARet
CATalog
CB
CBUS
CENTER
CENTered
CH1
CH14
CH2
CH3
CH4
CHANNELMAPping
CHANnel
CHAR
CHARacter
CHECKSUMDR
CHECKsumerror
CHKSUMError
CHecksum
CHx
CJTP
CLAss
CLEAR
CLOCKCoax
CLOCKSymmetrical
CLOck
CMDBatch
CMEan
CMI
CMOFFSet
CODe
COLOROFF
COLORON
COLOr
COLUMN
COMMAND
COMMunication
COMPAT
COMPLetion
COMPleted
COMbination
COMmonmode
COMpare
CONDition
CONNection
CONStant
CONSumer
CONTRol
CONTention
COPy
COUNt
COUPling
COUnter
CQ
CQ0
CR
CRC
CRC16
CRC5
CRCHEADer
CRCTRAILer
CREATE
CRMs
CROSSHair
CSActive
CSLevel
CSMode
CSPLIT
CSSource
CURSor
CURVENext
CURVEStream
CURVe
CUSTOMPct
CUSTom
CWD
CYCLECount
CYCLEcount
D
D0
D1
D10
D11
D12
D13
D14
D15
D2
D3
D4
D5
D6
D7
D8
D9
DALL
DASHed
DATA
DATA0
DATA1
DATA2
DATABits
DATAPacket
DATASource
DATE
DATa
DAYS
DB
DBCA
DBM
DC
DCHAN
DCREJect
DCSLONGRead
DCSLONGWrite
DCSR
DCSRR2
DCSSRR1
DCSSW0
DCSSW1
DDHHMMSS
DDR
DDR1066
DDR1200
DDR1333
DDR1466
DDR1600
DDR1866
DDR2
DDR200
DDR2133
DDR2400
DDR266
DDR2666
DDR3
DDR3200
DDR333
DDR370
DDR3L
DDR4
DDR400
DDR4000
DDR4800
DDR5000
DDR533
DDR5500
DDR667
DDR800
DDRMemory
DDRREADWrite
DDRRead
DDRWrite
DECOde
DECimal
DEFault
DEFine
DEGAUSS

DEGrees
DELETEAREA
DELEte
DELIMiter
DELTATime
DELTa
DELay
DELayed
DESE
DESKew
DEST
DESTination
DESTinationaddr
DESTinationport
DETAiled
DEViation
DIAg
DIFFerential
DIGITALALLQString
DIGItal
DIGTraces
DIGital
DIR
DIRection
DISPCOUNT
DISPRATE
DISParity
DISTDUty
DISable
DISplay
DISplaymode
DIVisions
DMINus
DMOFFSet
DONE
DONTcare
DOTs
DPLUs
DREF
DS0Contra
DS0Double
DS0Single
DS0Timing
DS1
DS1A
DS1C
DS2
DS2RATECoax
DS2RATESymmetrical
DS3
DS4NA
DS4NA_Max
DSINR
DSIVC
DSIViolation
DTLong
DTShort
DUAL
DURation
DYNAMIC
Dx
E1
E1Coax
E1Symmetrical
E2
E3
E4
E4_0
E4_1
ECCError
ECCMBError
ECCSBError
ECCWarn
ECL
EDGE
EDGE2
EDGE[1]
EEPROm
EI
EIE
EIGHtytwenty
EITher
EMBEDded
EMBOss
EMail
ENAble
ENCdg
END
ENDPACKET
ENDPoint
ENDevent
ENET100
ENET1000BCX_ATP2
ENET1000BCX_ATP3
ENET1000BCX_NTP2
ENET100BASETX
ENET100FX
ENET100STP
ENET100UTP
ENET10BASET
ENET1250
ENETXAUI
ENETXAUI2
ENETXAUI_Far
ENETXAUI_Near
ENGRave
ENHANCEDEnob
ENHanced
ENTERSWindow
ENV
ENVelope
ENable
EOF
EOFTYPE
EOP
EOT
EOTSync
EOp
EQUal
EQual
ERR
ERROR
ERRORLIMIT
ERRORdetector
ERRTYPE
ERRor
ESCMODEError
ESCMode
ESCape
ET
ETHernet
EVEN
EVENT
EVENTS
EVENTtable
EVMsg
EVQty
EXECute
EXITSWindow
EXPort
EXTAtten
EXTDBatten
EXTENded
EXTINCTDB
EXTINCTPCT
EXTINCTRATIO
EXTUnits
EXTernal
EYE
EYEHeight
EYEWIdth
EYEdiagram
FACtory
FAILURES
FAILure
FALL
FALSe
FALling
FAST
FASTAcq
FASTERthan
FASTframe
FAStest
FC1063
FC1063Draft
FC1063E
FC1063E_ABR
FC1063E_ABT
FC1063E_ADR
FC1063E_ADT
FC1063E_AGR
FC1063E_AGT
FC1063E_NBT
FC1063E_NDT
FC1063E_NGT
FC133
FC133E
FC2125
FC2125E_ABR
FC2125E_ABT
FC2125E_ADR
FC2125E_ADT
FC2125E_AGR
FC2125E_AGT
FC2125E_NBT
FC2125E_NDT
FC2125E_NGT
FC266
FC266E
FC4250
FC4250E_ABR
FC4250E_ABT
FC4250E_ADR
FC4250E_ADT
FC4250E_AGR
FC4250E_AGT
FC4250E_NBT
FC4250E_NDT
FC4250E_NGT
FC531
FC531E
FCE
FCSError
FFWD
FIELD
FIELD1
FIELD2
FIFtyfifty
FILEFormat
FILEName
FILESystem
FILTer

FILe
FILepath
FIRst
FIVe
FLATTOP2
FLEXRAY
FLIP
FOCUS
FONTCOlor
FONTEFfect
FONTNAme
FONTSIze
FONTSTyle
FORCEDRange
FORCESAMEFilesize
FORCe
FORMat
FORWard
FORWards
FP
FPANel
FPBinary
FRAME
FRAMEEnd
FRAMEID
FRAMELock
FRAMENUMber
FRAMESTOP
FRAMEStart
FRAMEtype
FRAMING
FRAme
FREE
FREQuency
FREV
FROMSearchx
FROm
FST
FST1
FST2
FST3
FST4
FST5
FST6
FTS
FULL
FULLNOmenu
FULLSCREEN
FULLSPeed
FULl
FUNCtion
FW1394BS1600B
FW1394BS1600BT1
FW1394BS1600BT2
FW1394BS400B
FW1394BS400BT1
FW1394BS400BT2
FW1394BS800B
FW1394BS800BT1
FW1394BS800BT2
FWD
FuLl
G703DS1
G703DS3
GAIN
GAIn
GATEPOS
GATEWIDTH
GATing
GAUSSian
GDDR3
GENeralcall
GLItch
GLONGRead
GLONGWrite
GND
GPIBUsb
GR0
GR1
GR2
GRAticule
GREATerthan
GREen
GRId
GROUPDelay
GSP1
GSP2
GSP3
GSP4
GSP5
GSP6
GSP7
GSP8
GSRR1
GSRR2
GSW0
GSW1
GSW2
HALT
HAMMing
HANDSHAKEPacket
HANDSHAKEType
HANDShaketype
HANNing
HARDCopy
HBArs
HBWStraightflex
HD1080I50
HD1080I60
HD1080P24
HD1080P25
HD1080P30
HD1080P50
HD1080P60
HD1080SF24
HD480P60
HD576P50
HD720P30
HD720P50
HD720P60
HD875I60
HDB3
HDELTA
HDR
HEADer
HEIght
HERtz
HEX
HEXAgon
HEXadecimal
HFRej
HFTP
HIACQRATE
HIERarchical
HIGH
HIGHLIGHTHits
HIGHLOWX
HIGHLimit
HIRes
HIStogram
HITs
HIVALue
HLS
HOLDTime
HOLdoff
HORizontal
HOSTwanted
HOURS
HPOS
HSCAle
HSMODe
HSRTError
HST
HST1
HST2
HST3
HST4
HST5
HST6
HSYNCEnd
HSYNCStart
HTRIGPOS
HUB
HWVersion
HYSteresis
I2C
ICAPture
ID
IDANDDATA
IDFORmat
IDLE
IDLETime
IDNMultiscope
IDentifier
IMAGEView
IMAGe
IMMEDSTAte
IMMed
IN
INDBits
INDependent
INF2_5G
INF2_5GE
INFINIBAND
INFInite
INFPersist
INHERIT
INIT
INITIALCRCVALue
INKSaver
INPUT
INPUTMode
INPut
INSIDEGreater
INSTR
INSide
INTENSIFied
INTENSITy
INTERLAced
INTERPEightbit
INTERPRatio
INTERRUPT
INTERnal
INVALID
INVert
INVerted
INrange
IPHeader
IPV4Header
IRE
ISCLOCKED
ISOALL

ISOEND
ISOMID
ISOSTART
ISOchronous
IT
ITEM
JPEG
KAISERBessel
LABel
LANE
LANdscape
LARge
LAST
LAYout
LBPT
LEARN
LENgth
LESSEQual
LESSLimit
LESSthan
LEVel
LF
LFRej
LFTP
LIMITrecordlen
LIMit
LIN
LINE
LINEAr
LINEEnd
LINEPeriod
LINES
LINESTyle
LINEStart
LINE_X
LISt
LIVE
LMPPACKets
LOCKLen
LOCKOffset
LOG
LOGAction
LOGIc
LOGNumber
LOGTable
LOGic
LONG
LONGORDERedset
LOOP
LOOPS
LOW
LOWLimit
LOWPASSfilter
LOWSPeed
LOWTHRESHold
LOWerthreshold
LOck
LPDATA
LPDDR
LPDDR3
LPS666
LPTSError
LSB
LVLSrcpreference
MAC
MACADDRess
MACLENgth
MACTYPE
MAG
MAGnivu
MAIn
MANChester
MANual
MARK
MARKALLevents
MARKS
MARKSINCOLumn
MARgin
MASK
MASKHighlight
MASKPRE
MATH1
MATH14
MATH2
MATH3
MATH4
MATHCOLOr
MATHCad
MATHVAR
MATHx
MATHy
MATLab
MAXALIGNS
MAXCas
MAXFRames
MAXRETsize
MAXSize
MAX_AP
MAXimum
MDATA
ME
MEAN
MEANSTDdev
MEAS1
MEAS2
MEAS3
MEAS4
MEAS5
MEAS6
MEAS7
MEAS8
MEASUrement
MEDian
MEDium
METHod
MFTP
MHZ10
MHZ100
MID
MIL1553B
MINCas
MINIMIZED
MINMax
MINUS
MINUSOne
MINUTES
MINimum
MINus
MIPICSITWo
MIPIDSIOne
MISO
MISOMOSI
MIXed
MKDir
MLT3
MODe
MONOGRAY
MONOGREEN
MOREEQua
MOREEQual
MORELimit
MOREthan
MOSI
MRTTime
MSB
MULTiscope
MULtipleframes
MV
NAK
NAME
NAMe
NANd
NCROss
NDUty
NEGAtive
NEWpass
NEXT
NO
NOCARE
NOISErej
NOISe
NONe
NOPARity
NOR
NORMALIZed
NORMal
NOVershoot
NR1
NR1NR1
NR1NR1NR1
NR1OFF
NR1QString
NR2
NR3
NRF
NRZ
NR_FR
NR_Pt
NTSC
NTSc
NUL1
NULL
NULLFRDYNAMIC
NULLFRSTATIC
NUMACq
NUMAVg
NUMEMails
NUMERic
NUMEnv
NUMEvents
NUMFRAMESACQuired
NUMFRames
NUMITEMS
NUMSAMples
NUMSymbols
NUMX
NUMY
NUMber
NWIdth
NYET
No
OC1
OC12
OC3
OC48
OC48_FEC
OCCurs
ODD
OFF
OFFGRAticule
OFFSETAdj
OFFset

ON
ONE
ONEfifty
ONGRAticule
ONLYOVERlay
ONOFF
OPCEXtended
OPTion
OR
ORDERedset
ORR
OTHer
OUT
OUTSIDEGreater
OUTrange
OUTside
OVERLoad
OVERlay
OWNer
PACKET
PACKetdata
PACKets
PAL
PALEtte
PARItyerror
PARallel
PARity
PASS
PASSWord
PATTERNBITS
PATTERNNAME
PATtern
PAYLENgth
PAYload
PBASe
PCIE
PCIE1_COMP_BIT
PCIE2_COMP_BIT
PCIEGEN1
PCIEGEN2
PCIEXPRESS_Rcv
PCIEXPRESS_Xmit
PCIExpress
PCIExpress2
PCROss
PCTCROss
PCX
PDUty
PEAKHits
PEAKdetect
PENDING
PERCent
PERIod
PERSistence
PERiod
PHASe
PID
PING
PIXel
PK2Pk
PKPKJitter
PKPKNoise
PLOCK
PLOCK BITRate
PLUS
PLUSOne
PNG
POINTS
POLARCoord
POLarity
PORT
PORTRait
POSITIVe
POSition
POVershoot
PPS101010
PPS121212
PPS565
PPS666
PPS888
PRBS7
PRBS9
PRE
PRESAMPBITS
PRESS
PREVious
PREscaler
PREset
PRINTER
PRInt
PROBECOntrol
PROBECal
PROBEFunc
PROBe
PRODDELta
PRODUCTID
PRODucer
PRODuct
PROGressive
PROTOcol
PRObe
PRObestate
PTOP
PT_Fmt
PT_ORder
PT_Off
PULSEForm
PULse
PWIdth
QFACtor
QString
QStringQString
QTAG
QUALification
QUALifier
QUALify
Qstring
QstringQstring
RADians
RANDOM
RANDom
RANge
RATDELta
RATE
RATE1
RATE100K
RATE10K
RATE10M
RATE1250
RATE125K
RATE1500
RATE153K
RATE19K
RATE1K
RATE1M
RATE2
RATE20K
RATE2125
RATE2500
RATE250K
RATE25K
RATE2K
RATE2M
RATE3
RATE3000
RATE3125
RATE31K
RATE32Mbit
RATE33K
RATE37K
RATE4
RATE400K
RATE4250
RATE4K
RATE5
RATE5000
RATE500K
RATE50K
RATE5M
RATE6000
RATE6250
RATE62K
RATE68K
RATE8000
RATE800K
RATE83K
RATE92K
RATE97Mbit
RATE9K
RATEcustomCUSTOM
RATIOENable
RATio
RAW
RAW10
RAW12
RAW14
RAW6
RAW7
RAW8
RDELta
RDMINUS
RDPLUS
READ
READFile
READOUT
READOuts
READY
RECAll
RECORDView
RECOVered
RECOrdlength
RECTangle
RECTangular
RED
REF
REF1
REF14
REF2
REF3
REF4
REFCOLOr
REFClk
REFLEVELOffset
REFLevel
REFOUT
REFerence
REFx
REJECTCOUNT
REJECTTIMEout
REJect
REJects
RELoad

REM
REMote
REName
REPEATstart
REPeat
RESBw
RESERVED
RESETLive
RESPonsetime
RESULT
RESUME
RESet
RESistance
RESults
REV
REVErse
RGB
RI
RIBinary
RIO_1G
RIO_1_5G
RIO_2G
RIO_500M
RIO_750M
RIO_DRV1G
RIO_DRV1_5G
RIO_DRV2G
RIO_DRV500M
RIO_DRV750M
RIO_EDRV1G
RIO_EDRV1_5G
RIO_EDRV2G
RIO_EDRV500M
RIO_EDRV750M
RIO_RCV1G
RIO_RCV1_5G
RIO_RCV2G
RIO_RCV500M
RIO_RCV750M
RIO_SERIAL_1G
RIO_SERIAL_2G
RIO_SERIAL_2_5G
RIO_SERIAL_3G
RISe
RISetime
RISing
RMDir
RMS
RMSJitter
RMSNoise
ROLL
ROLe
ROSc
ROTAtion
RP
RPBinary
RS232
RS232c
RT
RUN
RUNSTop
RUNT
RWINADDR
RWINClude
RX
S8B10B
SAMPLERate
SAMPlingmode
SAMple
SAS1_5
SAS1_5_CR
SAS1_5_CR_AASJ
SAS1_5_IR
SAS1_5_IR_AASJ
SAS1_5_SATA
SAS1_5_XR
SAS1_5_XR_AASJ
SAS3_
SAS3_0_CR
SAS3_0_CR_AASJ
SAS3_0_IR
SAS3_0_IR_AASJ
SAS3_0_SATA
SAS3_0_XR
SAS3_0_XR_AASJ
SAS6_0
SATA1_5
SATA1_CJTPAT_BIT
SATA2_CJTPAT_BIT
SATA3_0
SATA3_CHAR
SATA3_CJTPAT_BIT
SATA3_FRAME
SATA3_HFTP_BIT
SATA3_LBP_BIT
SATA3_LFTP_BIT
SATA3_MFTP_BIT
SATA6_0
SATAGEN1
SATAGEN2
SATAGEN3
SAVE
SAVEIMAGE
SAVEON
SAVEWFM
SAVe
SC
SCAN
SCAle
SCLK
SCOPEAPP
SCRAMBLED
SCREEN
SCREENSAVER
SCREENSAVERDELAY
SCREENTExt
SCROLL
SCROLLLock
SDASHed
SE
SEARCH
SEARCHtotrigger
SECAM
SECOnds
SELECTED
SELECtoffset
SELect
SENDEMAIL
SENSITivity
SEQUENTIAL
SEQnum
SEQuence
SERIAL
SERIALTRIG
SERIALnumber
SERnumber
SET
SETCHANGE
SETHold
SETLOGIC
SETLevel
SETROLE
SETSTATE
SETTime
SETUp
SFD
SFI5_2
SFI5_3
SFI5_RCVBCLK2
SFI5_RCVBCLK3
SFI5_RCVBDATA2
SFI5_RCVBDATA3
SFI5_RCVDCLK2
SFI5_RCVDCLK3
SFI5_RCVDDATA2
SFI5_RCVDDATA3
SFI5_XMITACLK2
SFI5_XMITACLK3
SFI5_XMITADATA2
SFI5_XMITADATA3
SFI5_XMITCCLK2
SFI5_XMITCCLK3
SFI5_XMITCDATA2
SFI5_XMITCDATA3
SFPbinary
SHADow
SHAPE
SHIFted
SHORt
SHOWEQuation
SHOWLOGic
SHOWLog
SHOWMeasurement
SHOWREmote
SHUTDown
SIGMA1
SIGMA2
SIGMA3
SIGNAL
SIGType
SIGnal
SIGnaltype
SINGLEFramemath
SINGleended
SINX
SIXSigmajit
SIXteenbit
SIZe
SKIPSETPRIMitive
SKP
SLEEP
SLOWERthan
SLOpe
SMAll
SMTPPort
SMTPServer
SNAp
SNRatio
SOF
SOFFRAMENUMber
SOLID
SOT
SOTError
SOTSync
SOURCES
SOUrce
SOUrceaddr
SOUrceport
SPAN
SPC
SPECIALPacket
SPECIALType

SPECTral
SPEed
SPI
SPLit
SPREADSHEETCsv
SPREADSHEETTxt
SPace
SRCDependent
SRCIndependent
SRIbinary
SRPbinary
SRQ
SS
SSC
SSPLIT
STABle
STALL
STANdard
START
STARTFRAME
STARTUPNOSYNC
STARTevent
STARTup
STARt
STARtbyte
STATE
STATIC
STATIstics
STATus
STAYSHigh
STAYSLow
STAte
STDdev
STM0_0
STM0_1
STM0_CMI
STM0_HDBX
STM1E
STM1E_0
STM1E_1
STOP
STOPAfter
STOPOnviolation
STOPWHEN
STORe
STROBE
STROBEsource
STRing
STS1
STS1Eye
STS1Pulse
STS3
STS3_Max
STYle
STandard
SUBADdress
SUBITEMS
SUBSF
SUBSYS
SUMFrame
SUPPress
SUSPEND
SYMBOLS
SYMBolic
SYMbol
SYNC
SYNCFRAME
SYNCInterval
SYNCPATtern
SYNCSOUrces
String
TCPDATA
TCPData
TCPHeader
TEAR
TEKEXPonential
TEKLink
TEKSecure
TEMPErature
TEMPlate
TENbit
TERmination
TESTS
TESt
TF
TFI5_2
TFI5_3
THDELta
THReshold
TIFF
TIME
TIMEOut
TIMEStamp
TIMEformat
TIMe
TIPtype
TO
TOGGLE
TOKENPacket
TOKENType
TOLerance
TOTALCOUNT
TOTal
TPPACKets
TRACk
TRACking
TRANsition
TRAPezoid
TRBit
TRIAngle
TRIG
TRIGBar
TRIGIF
TRIGT
TRIGTOSAMP
TRIGger
TRIGgeron
TRIGgertosearch
TRILevelcustom
TRIgger
TRUe
TS1
TS2
TTL
TURNON
TWEnty
TWOfifty
TX
TXRX
TYPE
TYPe
U
ULTRALP
ULTRAsync
UNDEFINED
UNDo
UNEQual
UNITString
UNIts
UNLock
UNWRap
UPDate
UPPerthreshold
USB
USB3
USB3_CHAR
USB3_SYMBOL
USBTMC
USER
USERMask
USEr
V
V1X
V2X
VALUEMean
VALUEOne
VALUETwo
VALue
VARpersist
VBArs
VC
VDELTA
VECtors
VENDORID
VERBose
VERTICES
VERT_INDEX
VERTical
VFields
VIDEO270
VIDEO292M
VIDEO360
VIDeo
VIEW
VIOLATIONS
VISual
VLines
VOFFSet
VPOS
VSCAle
VSROC192
VSYNCEnd
VSYNCStart
VTERm
WA
WAIT
WAKEup
WARNing
WAVEFORMS
WAVEform
WAVFRMStream
WAVFrm
WC
WEIghting
WFId
WFMCH1
WFMCH2
WFMCH3
WFMCH4
WFMDB
WFMInpre
WFMOutpre
WFMPre
WHEn
WIDERthan
WIDth
WINdow
WIThin
WRITE
WRITEFile
X
XDELta

XFF
XINcr
XLARGE
XPOS
XPOSition
XSMAll
XUNit
XY
XYZ
XZEro
Y
YCBCR
YCBCR12
YCBCR16
YCBCR20
YCBCR24
YDELta
YES
YMUlt
YOFf
YPOS
YPOSition
YT
YUNit
YUV
YUV420B8
YUV420C8
YUV420L8
YUV422B10
YUV422B8
YZEro
Yes
ZERo
ZOOMx
ZOOm
asccurve
decodeFileName
directorypath
dynamicRangeNR3
fileName
filepath
filepathGPIb
filepathREFx
filepathdata
high
low
menuoff
newworkingdir
nr1
nr2
nr3
oldfilepathnew
rangeName
string
wfm
wfmREFx
wfmwfm
x

Appendix C: Factory default setup values

Default setup

The following tables list the default setup values by command group. The instrument sets these values when you press the front panel DEFAULT SETUP button or send the FACtory command. Only those commands that have values set by the DEFAULT SETUP function are listed in these tables.

NOTE. *The following commands do not affect the DATa parameters: TEKSecure, *RST, RECALL:SETUP FACtory. These are the initial factory default settings for DATa :DATA:DESTINATION REF1;ENCDG RIBINARY;SOURCE CH1;START 1;STOP 5000;FRAMESTART 1;FRAMESTOP 5000.*

NOTE. *Find the most up-to-date values for your instrument and software by pressing default setup and looking at the instrument to get the default values.*

NOTE. *Some of these commands may not be available on your model.*

Table C-1: Acquisition default values

Command	Default values
ACQuire:MAGnivu	OFF
ACQuire:MODe	SAMPLE
ACQuire:NUMAVg	16
ACQuire:NUMEnv	10
ACQuire:NUMSAMples	100000
ACQuire:SAMPlingmode	IT
ACQuire:STATE	1 (ON)
ACQuire:STOPAfter	RUNSTOP
FASTAcq:STATE	0

Table C-2: Alias default values

Command	Default values
ALIas:STATE	0 (OFF)

Table C-3: Bus default values (Not available on some models)

Command	Default values
BUS:B<x>:CAN:SOUrce	CH1
BUS:B<x>:CAN:BITRate	RATE500k

Table C-3: Bus default values (Not available on some models) (cont.)

Command	Default values
BUS:B<x>:CAN:BITRate:VALue	500.0000E+3
BUS:B<x>:CAN:PRObe	CANH
BUS:B<x>:DISplay:DECOde:STAte	OFF
BUS:B<x>:FLEXRAY:BITRate	RATE10M
BUS:B<x>:FLEXRAY:BITRate:VALue	10.0000E+6
BUS:B<x>:FLEXRAY:CHANnel	A
BUS:B<x>:FLEXRAY:SOUrce	CH1
BUS:B<x>:FLEXRAY{:PROBe|:SIGnal}	BDIFFBP
BUS:B<x>:I2C:CLOCk:SOUrce	D1
BUS:B<x>:I2C:DATa:SOUrce	D0
BUS:B<x>:I2C:RWINADDR	No
BUS:B<x>:LABel	“Bus 1”
BUS:B<x>:LIN:BITRate	RATE19K
BUS:B<x>:LIN:BITRate:VALue	19.2000E+3
BUS:B<x>:LIN:IDFORmat	NOPARity
BUS:B<x>:LIN:POLarity	NORMal
BUS:B<x>:LIN:SOUrce	CH1
BUS:B<x>:LIN:STANDard	MIXed
BUS:B<x>:PARallel:CLOCk:EDGE	Rising
BUS:B<x>:PARallel:CLOCk:SOUrce	D15
BUS:B<x>:PARallel:ISCLOCKED	No
BUS:B<x>:PCIE:BITRate	RATE2500
BUS:B<x>:PCIE:BITRate:VALue	2.5Gb/s
BUS:B<x>:PCIE:HYSTeresis	50mV
BUS:B<x>:PCIE:LANE	0
BUS:B<x>:PCIE:SOUrce	CH1
BUS:B<x>:POSition	0.0 div
BUS:B<x>:RS232C:BITRate	9600 bits-per-second
BUS:B<x>:RS232C:DATABits	8
BUS:B<x>:RS232C:DISplaymode	Frame
BUS:B<x>:RS232C:PARity	None
BUS:B<x>:RS232C:POLarity	Normal
BUS:B<x>:RS232C:SOUrce	Ch1
BUS:B<x>:S8B10B:BITRate	1250
BUS:B<x>:S8B10B:BITRate:VALue	1.2500E+9
BUS:B<x>:S8B10B:HYSTeresis	20.0000E-3
BUS:B<x>:S8B10B:SOUrce	CH1

Table C-3: Bus default values (Not available on some models) (cont.)

Command	Default values
BUS:B<x>:SPI:BITOrder	MSB
BUS:B<x>:SPI:CLOCk:POLarity	Rising
BUS:B<x>:SPI:CLOCk:SOUrce	D2
BUS:B<x>:SPI:DATa:POLarity	Normal
BUS:B<x>:SPI:DATa:SIZe	8
BUS:B<x>:SPI:DATa:SOUrce	D0
BUS:B<x>:SPI:FRAMING	SS
BUS:B<x>:SPI:IDLETime	5.0000E-6
BUS:B<x>:SPI:SELect:POLarity	Active low
BUS:B<x>:SPI:SELect:SOUrce	D1
BUS:B<x>:TYPe	Parallel
BUS:B<x>:USB:BITRate	Full
BUS:B<x>:USB:PRObe	Differential
BUS:B<x>:USB:SOUrce	Ch1
BUS:B<x>:USB:SOUrce:DMINus	Ch1
BUS:B<x>:USB:SOUrce:DPLUs	Ch1
SELect:B<x>	OFF

Table C-4: Cursor default values

Command	Default values	
CURSor:FUNCtion	VBARS	
CURSor:HBArs:POSITION<x>	1 = 300.0000E-3	2 = -300.0000E-3
CURSor:LINESTyle	SDASHED	
CURSor:MODe	INDEPENDENT	
CURSor:SCREEN:STYle	LINES	
CURSor:SOUrce<x>	CH1	
CURSor:STATE	0	
CURSor:VBArs:POSITION<x>	**For instruments ≥12.5 GHz**	**For instruments ≤3.5 GHz**
	1 = –8.0000E-9	1 = –40.0000E-9
	2 = 8.0000E-9	2 = 40.0000E-9
	For instruments >3.5 GHz and <12.5 GHz	
	1 = –16.0000E-9	
	2 = 16.0000E-9	
CURSor:VBArs:UNIts	SECONDS	

Table C-4: Cursor default values (cont.)

Command	Default values	
CURSor:WAVEform:POSition<x>	**For instruments ≥12.5 GHz**	**For instruments ≤3.5 GHz**
	1 = –8.0000E-9	1 = –40.0000E-9
	2 = 8.0000E-9	2 = 40.0000E-9
	For instruments >3.5 GHz and <12.5 GHz	
	1 = –16.0000E-9	
	2 = 16.0000E-9	
CURSor:WAVEform:STYle	LINE_X	
CURSor:WAVEform:UNIts	BASE	
CURSor:XY:READOUT	RECTANGULAR	
CURSor:XY:RECTX<x>	0.0000	
CURSor:XY:RECTY<x>	0.0000	

Table C-5: Diagnostics default values

Command	Default values
DIAg:LEVEL	SUBSYS

1 The factory default is not restored by `Default Setup`, `TekSecure`, or `*RST`.

Table C-6: Digital default values (Not available on some models)

Command	Default values
DISplay:DIGital:HEIght	Large
D<x>:POSition	–4.66 div
D<x>:THRESHold	1.4 V
REF<x>:THRESHold	1.0 mV

Table C-7: Display control default values

Command	Default values	
DISplay:CLOCk	1	
DISplay:COLOr:MATHCOLOr	DEFAULT	
DISplay:COLOr:PALEtte:IMAGEView [1]	TEMPERATURE	
DISplay:COLOr:PALEtte:RECORDView	NORMAL	
DISplay:COLOr:PALEtte:USEr:CARet	150, 50, 100	
DISplay:COLOr:PALEtte:USEr:CH<x> [1]	1 = 180, 50, 100	3 = 60, 50, 100
	2 = 300, 50, 100	4 = 240, 50, 100
DISplay:COLOr:PALEtte:USEr:GRAticule [1]	On DPO7000 series of products it is 320, 10, 0	
	On MSO70000 series of products it is 165, 50, 15	

Table C-7: Display control default values (cont.)

Command	Default values	
DISplay:COLOr:PALEtte:USEr:HIStogram [1]	On DPO7000 series of products it is 320,10, 0 On MSO70000 series of products it is 320, 50,100	
DISplay:COLOr:PALEtte:USEr:MASK [1]	On DPO7000 series of products it is 320, 10, 0 On MSO70000 series of products it is 0, 40, 100	
DISplay:COLOr:PALEtte:USEr:MASKHighlight [1]	On DPO7000 series of products it is 320,10, 0 On MSO70000 series of products it is 140, 50, 100	
DISplay:COLOr:PALEtte:USEr:MATH<x>	1 = 160, 50, 100 2 = 40, 60, 100	3 = 120, 60, 100 4 = 195, 50, 100
DISplay:COLOr:PALEtte:USEr:REF<x>	1 = 0, 90, 0 2 = 0, 90, 100	3 = 60, 90, 100 4 = 240, 90, 100
DISplay:COLOr:REFCOLOr	DEFAULT	
DISplay:FILTer	SINX	
DISplay:FORMat	YT	
DISplay:GRAticule	FULL	
DISplay:INTENSITy:BACKLight	High	
DISplay:INTENSITy:WAVEform:IMAGEView	75.0000	
DISplay:INTENSITy:WAVEform:RECORDView	75.0000	
DISplay:PERSistence	OFF	
DISplay:SCREENTExt:LABel<x>:NAMe	""	
DISplay:SCREENTExt:LABel<x>:XPOS	100	
DISplay:SCREENTExt:LABel<x>:YPOS	1 = 5 2 = 20 3 = 35 4 = 50	5 = 65 6 = 80 7 = 95 8 = 110
DISplay:SCREENTExt:STATE	0	
DISplay:STYle	VECTORS	
DISplay:TRIGBar	SHORT	
DISplay:TRIGT	0	
DISplay:VARpersist	500.0000E-3	

[1] The factory default is not restored by Default Setup, TekSecure, Factory, or *RST. The factory default is restored by the `DISplay:COLOr:PALEtte:USEr` Reset command.

Table C-8: Email default values

Command	Default values
EMail:ATTempts [1]	1
EMail:AUTHLogin	""
EMail:FROm [1]	"DPO7104_<Instrument Serial Number>"
EMail:HOSTwanted [1]	""

Table C-8: Email default values (cont.)

Command	Default values
EMail:IMAGe	0
EMail:MASK	0
EMail:MAXSize	1
EMail:MEASUrement	0
EMail:NUMEMails	1
EMail:SMTPPort [1]	25
EMail:SMTPServer	""
EMail:TIMEOut [1]	30
EMail:TO	""
EMail:TRIGger	0
EMail:WAVEform	0

1 These factory defaults are not reset to factory defaults by Default Setup.

Table C-9: Error detector default values (Not available on some models)

Command	Default values
ERRORDetector:ALERT	OFF
ERRORDetector:ALIGNCHARacter:MINus?	“0011111010"
ERRORDetector:ALIGNCHARacter:PLUS?	“1100000101"
ERRORDetector:ALIGNCHARacter:SYMBOL	“K28.5”
ERRORDetector:ALIGNPRIMitive:MINUS?	“0011111010”,”0101010101”,”0101010101”,”0010011100”
ERRORDetector:ALIGNPRIMitive:PLUS?	“1100000101”,”0101010101”,”0101010101”,”1101100011”
ERRORDetector:ALIGNPRIMitive:STATE	ON
ERRORDetector:ALIGNPRIMitive:SYMBOL<x>	“K28.5”,”D10.2”,”D10.2”,”D27.3”
ERRORDetector:BIT:LENgth	0
ERRORDetector:BIT:SYNCPATtern:ADVanced	OFF
ERRORDetector:BIT:SYNCPATtern:BITString	“0011111010” (RD- K28.5)
ERRORDetector:BIT:SYNCPATtern:DISParity<x>	RDPLUS, RDMINUS, RDPLUS, RDMINUS
ERRORDetector:BIT:SYNCPATtern:MINus<x>?	“0011111010”,”XXXXXXXXXX”,”XXXXXXXXXX”,”XXXXXXXXXX”
ERRORDetector:BIT:SYNCPATtern:PLUS<x>?	“1100000101”,”XXXXXXXXXX”,”XXXXXXXXXX”,”XXXXXXXXXX”
ERRORDetector:BIT:SYNCPATtern:SYMBOL<x>	“K28.5”,”X”,”X”,”X”
ERRORDetector:BIT:SYNCPATtern?	K28.5, “0011111010”, and “1100000101”
ERRORDetector:BITRate	RATE6000 (6 Gb/s)
ERRORDetector:BITRate:VALue	6000000000 (6Gb/s)
ERRORDetector:CHANnel	CH1
ERRORDetector:DURATION:COUNt	-1 (count forever)
ERRORDetector:DURATION:SECOnds	60 (total time in seconds)
ERRORDetector:DURATION:TIME:DAYS	0

Table C-9: Error detector default values (Not available on some models) (cont.)

Command	Default values
ERRORDetector:DURATION:TIME:HOURS	0
ERRORDetector:DURATION:TIME:MINUTES	1
ERRORDetector:DURATION:TIME:SECOnds	0
ERRORDetector:ERRORLIMIT	4
ERRORDetector:FONTSIze	LARGE
ERRORDetector:FRAme:INITIALCRCVALue	SATA
ERRORDetector:MAXALIGNS	0
ERRORDetector:PATTERNNAME	CJTPat
ERRORDetector:PREset	SATA3_CJTPAT_BIT
ERRORDetector:SAVEIMAGE	OFF
ERRORDetector:SAVEWFM	OFF
ERRORDetector:SCRAMBLED	ON
ERRORDetector:SENDEMAIL	OFF
ERRORDetector:SIGnaltype	SATAGEN3
ERRORDetector:SSC	ON
ERRORDetector:STANdard	SATA
ERRORDetector:STATE	OFF
ERRORDetector:STOPWHEN	MANUAL
ERRORDetector:TIMEformat	DDHHMMSS
ERRORDetector:TYPe	BIT

Table C-10: Hard copy default values

Command	Default values
HARDCopy:FILEName	""
HARDCopy:PORT	FILE

Table C-11: Histogram default values

Command	Default values
HIStogram:BOXPcnt	30.0000, 25.1000, 70.0000, 75.2000
HIStogram:DISplay	LINEAR
HIStogram:FUNCtion	HORIZONTAL
HIStogram:SIZe	2.0000
HIStogram:SOUrce	CH1
HIStogram:STATE	0

Table C-12: Horizontal default values

Command	Default values
HORizontal:ACQDURATION?	40 ns
HORizontal:ACQLENGTH?	1000
HORizontal:DIGital:RECOrdlength?	10000
HORizontal:DIGital:RECOrdlength:MAGnivu?	8250
HORizontal:DIGital:RECOrdlength:MAIn?	1000
HORizontal:DIGital:SAMPLERate?	25 GS/s
HORizontal:DIGital:SAMPLERate:MAGnivu?	16.5 GS/s
HORizontal:DIGital:SAMPLERate:MAIn?	500 MS/s ***NOTE.*** *Not available on some models.*
HORizontal[:MAIn]:DELay:MODe	0
HORizontal[:MAIn]:DELay:POSition	50.0000
HORizontal[:MAIn]:DELay:TIMe	0.00000000000
HORizontal[:MAIn]:POSition	50.0000
HORizontal:FASTframe?	OFF
HORizontal:FASTframe:COUNt	2
HORizontal:FASTframe:MAXFRames?	1
HORizontal:FASTframe:SINGLEFramemath	0
HORizontal:FASTframe:SIXteenbit	1
HORizontal:MAIn:UNIts	"s"
HORizontal:MAIn:UNIts:STRing	"s"
HORizontal:MODE	AUTO
HORizontal:MODE:AUTO:LIMITrecordlen	100000
HORizontal:MODE:RECOrdlength	≤2.5 GHz instruments: 1.0E+3
HORizontal:MODE:SAMPLERate	≥12.5 GHz instruments: 50 GS/s ≥3.5 GHz and ≤8 GHz instruments: 25.0 GS/s ≤3.5 GHz instruments: 10.0 GS/s
HORizontal:ROLL	AUTO

Table C-13: Low speed serial trigger default values

Command	Default values
TRIGger:A:BUS:CAN:CONDition	SOF
TRIGger:A:BUS:CAN:DATa:DIRection	DONTCare
TRIGger:A:BUS:CAN:DATa:FORMat	BINARY
TRIGger:A:BUS:CAN:DATa:QUALifier	EQUal
TRIGger:A:BUS:CAN:DATa:SIZe	1
TRIGger:A:BUS:CAN:DATa:VALue	XXXXXXXX
TRIGger:A:BUS:CAN:FRAMEtype	DATA

Table C-13: Low speed serial trigger default values (cont.)

Command	Default values
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:DIRection	DONTCare
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:FORMat	BINARY
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:MODe	STandard
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:VALue	XXXXXXXXX01
TRIGger:A:BUS:FLEXRAY:CONDition	SOF
TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat	BINary
TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier	EQUal
TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue	XXXXXX
TRIGger:A:BUS:FLEXRAY:DATa:FORMat	BINary
TRIGger:A:BUS:FLEXRAY:DATa:OFFSet	-1
TRIGger:A:BUS:FLEXRAY:DATa:QUALifier	EQUal
TRIGger:A:BUS:FLEXRAY:DATa:SIZe	1
TRIGger:A:BUS:FLEXRAY:DATa:VALue	XXXXXXXX
TRIGger:A:BUS:FLEXRAY:EOFTYPE	ALL
TRIGger:A:BUS:FLEXRAY:ERRTYPE	CRCHEADer
TRIGger:A:BUS:FLEXRAY:FRAMEType	NORMal
TRIGger:A:BUS:FLEXRAY:HEADER:CRC	XXXXXXXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount	XXXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID	XXXXXXXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:INDBits	XXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth	XXXXX
TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat	BINary
TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier	EQUal
TRIGger:A:BUS:FLEXRAY:IDentifier:VALue	XXXXXXXXXX
TRIGger:A:BUS:LIN:CONDition	SYNC
TRIGger:A:BUS:LIN:DATa:FORMat	BINary
TRIGger:A:BUS:LIN:DATa:QUALifier	EQUal
TRIGger:A:BUS:LIN:DATa:SIZe	1
TRIGger:A:BUS:LIN:DATa:VALue	XXXXXXXX
TRIGger:A:BUS:LIN:ERRTYPE	SYNC
TRIGger:A:BUS:LIN:IDentifier:FORMat	BINary
TRIGger:A:BUS:LIN:IDentifier:VALue	XXXXXX
TRIGger:A:BUS:PCIE:CHARacter:CHAR	COM
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus	0011111010
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS	1100000101
TRIGger:A:BUS:PCIE:CONDition	PATtern
TRIGger:A:BUS:PCIE:DISParity	EITher

Table C-13: Low speed serial trigger default values (cont.)

Command	Default values
TRIGger:A:BUS:PCIE:ERROR	CHARacter
TRIGger:A:BUS:PCIE:FORMat	ORDERedset
TRIGger:A:BUS:PCIE:PATtern:CHAR<x>	COM SKP SKP SKP for each of the 4 index elements
TRIGger:A:BUS:PCIE:PATtern:ORDERedset	SKP
TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>	0011111010 1100001011 1100001011 1100001011 for each of the 4 index elements
TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>	1100000101 0011110100 0011110100 0011110100 for each of the 4 index elements
TRIGger:A:I2C:ADDRess:MODe	10 Bit
TRIGger:A:I2C:ADDRess:RWINClude	OFF
TRIGger:A:I2C:ADDRess:TYPe	None
TRIGger:A:I2C:CONDition	Start
TRIGger:A:I2C:DATa:DIRection	don't care (X)
TRIGger:A:I2C:DATa:SOUrce	Ch 1
TRIGger:A:RS232:BAUd	19.2 kB
TRIGger:A:RS232:DATa:SOUrce	Ch 1
TRIGger:A:RS232:DATa:VALue	XXXX XX01
TRIGger:A:RS232:FORMat	Binary
TRIGger:A:RS232:PARity	None
TRIGger:A:SPI:CONDition	MOSI
TRIGger:A:SPI:DATa:MISO:ACTIVE	HIGH
TRIGger:A:SPI:DATa:MISO:SOUrce	Ch 1
TRIGger:A:SPI:DATa:MOSI:ACTIVE	HIGH
TRIGger:A:SPI:DATa:MOSI:SOUrce	Ch 1
TRIGger:A:SPI:SCLK:ACTIVE	Rising
TRIGger:A:SPI:SCLK:LEVel	0.0 V
TRIGger:A:SPI:SCLK:SOUrce	Ch 2
TRIGger:A:SPI:SS:ACTIVE	L
TRIGger:A:SPI:SS:LEVel	0.0 V
TRIGger:A:SPI:SS:SOUrce	Ch 3

Table C-14: Mask default values

Command	Default values
MASK:AUTOAdjust	10
MASK:AUTOAdjust:HDELTA	10
MASK:AUTOAdjust:VDELTA	10
MASK:AUTOSet:AUTOAdjust	1

Table C-14: Mask default values (cont.)

Command	Default values
MASK:AUTOSet:HPOS	1
MASK:AUTOSet:HSCAle	1
MASK:AUTOSet:MODe	MANUAL
MASK:AUTOSet:OFFSETAdj	1
MASK:AUTOSet:STANdard	NONE
MASK:AUTOSet:TRIGger	1
MASK:AUTOSet:VPOS	1
MASK:AUTOSet:VSCAle	1
MASK:COUNt:STATE	0
MASK:DISplay	1
MASK:FILTer	0
MASK:HIGHLIGHTHits	1
MASK:INVert	0
MASK:LOCk	1
MASK:MARgin:PERCent	5.0000
MASK:MARgin:STATE	0
MASK:MASKPRE:AMPlitude	100.0000E-3
MASK:MASKPRE:HSCAle	200.0000E-9
MASK:MASKPRE:HTRIGPOS	500.0000E-3
MASK:MASKPRE:PATTERNBITS	1
MASK:MASKPRE:PRESAMPBITS	0
MASK:MASKPRE:RECOrdlength	5000
MASK:MASKPRE:TRIGTOSAMP	0.0000
MASK:MASKPRE:VOFFSet	0.0000
MASK:MASKPRE:VPOS	0.0000
MASK:MASKPRE:VSCAle	1.0000E-3
MASK:MASKPRE:WIDth	0.0000
MASK:POLarity	POSITIVE
MASK:SOUrce	CH1
MASK:STANdard	NONE
MASK:STOPOnviolation	0
MASK:TESt:BEEP:COMPLetion	0
MASK:TESt:BEEP:FAILure	0
MASK:TESt:DELay	0.0000
MASK:TESt:HARDCopy	0
MASK:TESt:LOG:FAILure	0

Table C-14: Mask default values (cont.)

Command	Default values
MASK:TESt:REPeat	0
MASK:TESt:SAMple:THReshold	1
MASK:TESt:SAVEWFM	0
MASK:TESt:SAVEWFM:FILEName	“C:/TekScope/Waveforms/”
MASK:TESt:SRQ:COMPLetion	0
MASK:TESt:SRQ:FAILure	0
MASK:TESt:STATE	0
MASK:TESt:STOP:FAILure	0
MASK:TESt:THReshold	1
MASK:TESt:WAVEform	20
MASK:USER:AMPlitude	100.0000E-3
MASK:USER:BITRate	1544000
MASK:USER:HSCAle	200.0000E-9
MASK:USER:HTRIGPOS	500.0000E-3
MASK:USER:LABel	“User Mask”
MASK:USER:PATTERNBITS	1
MASK:USER:PRESAMPBITS	0
MASK:USER:RECOrdlength	5000
MASK:USER:TRIGTOSAMP	0.0000
MASK:USER:VOFFSet	0.0000
MASK:USER:VPOS	0.0000
MASK:USER:VSCAle	1.0000E-3
MASK:USER:WIDth	0.0000

Table C-15: Math default values

Command	Default values	
MATHVAR:VAR<x>	0.0000	
MATH<x>:DEFine	“ ”	
MATH<x>:LABel:NAMe	“”	
MATH<x>:LABel:XPOS	1.0000	
MATH<x>:LABel:YPOS	1 = 600.0000E-3	3 = -200.0000E-3
	2 = 200.0000E-3	4 = -600.0000E-3
MATH<x>:NUMAVg	2	
MATH<x>:SPECTral:CENTER	625.0000E+6	
MATH<x>:SPECTral:GATEPOS	0.0000	
MATH<x>:SPECTral:GATEWIDTH	2.0000E-6	
MATH<x>:SPECTral:LOCk	0	

Table C-15: Math default values (cont.)

Command	Default values
MATH<x>:SPECTral:MAG	DB
MATH<x>:SPECTral:PHASE	DEGREES
MATH<x>:SPECTral:REFLevel	20.0000
MATH<x>:SPECTral:REFLEVELOffset	223.6000E-3
MATH<x>:SPECTral:RESBw	1000.0000E+3
MATH<x>:SPECTral:SPAN	1.2500E+9
MATH<x>:SPECTral:SUPPress	–200.0000
MATH<x>:SPECTral:UNWRap	0
MATH<x>:SPECTral:WINdow	GAUSSIAN
MATH<x>:UNITString	“V”
MATH<x>:VERTical:POSition	0.0000
MATH<x>:VERTical:SCAle	1.0000

Table C-16: Measurement default values

Command	Default values
MEASUrement:GATing	OFF
MEASUrement:IMMed:DELay:DIREction	FORWARDS
MEASUrement:IMMed:DELay:EDGE[1]	RISE
MEASUrement:IMMed:DELay:EDGE2	RISE
MEASUrement:IMMed:METHod	HISTOGRAM
MEASUrement:IMMed:NOISe	HIGH
MEASUrement:IMMed:REFLevel:ABSolute:HIGH	0.0000
MEASUrement:IMMed:REFLevel:ABSolute:LOW	0.0000
MEASUrement:IMMed:REFLevel:ABSolute:MID<x>	0.0000
MEASUrement:IMMed:REFLevel:METHod	PERCENT
MEASUrement:IMMed:REFLevel:PERCent:HIGH	90.0000
MEASUrement:IMMed:REFLevel:PERCent:LOW	10.0000
MEASUrement:IMMed:REFLevel:PERCent:MID<x>	50.0000
MEASUrement:IMMed:SOUrce<x>	CH1
MEASUrement:IMMed:SOUrce<x>:SIGType	PULSE
MEASUrement:IMMed:TYPe	UNDEFINED
MEASUrement:MEAS<x>:DELay:DIREction	FORWARDS
MEASUrement:MEAS<x>:DELay:EDGE<x>	RISE
MEASUrement:MEAS<x>:METHod	HISTOGRAM
MEASUrement:MEAS<x>:NOISe	HIGH
MEASUrement:MEAS<x>:REFLevel:ABSolute:HIGH	0.0000

Table C-16: Measurement default values (cont.)

Command	Default values
MEASUrement:MEAS<x>:REFLevel:ABSolute:LOW	0.0000
MEASUrement:MEAS<x>:REFLevel:ABSolute:MID<x>	0.0000
MEASUrement:IMMed:REFLevel:METHod	PERCENT
MEASUrement:MEAS<x>:REFLevel:PERCent:HIGH	0.0000
MEASUrement:MEAS<x>:REFLevel:PERCent:LOW	0.0000
MEASUrement:MEAS<x>:REFLevel:PERCent:MID<x>	0.0000
MEASUrement:MEAS<x>:SOUrce<x>	CH1
MEASUrement:MEAS<x>:SOUrce<x>:SIGType	PULSE
MEASUrement:MEAS<x>:STATE	0
MEASUrement:MEAS<x>:TYPe	UNDEFINED
MEASUrement:STATIstics:MODe	ALL
MEASUrement:STATIstics:WEIghting	32

Table C-17: Miscellaneous default values

Command	Default values
APPLication:SCOPEAPP:WINDOW	RUNNING
AUXout:EDGE	FALLING
AUXout:SOUrce	ATRIGGER
CMDBatch	1 (ON)
HEADer	1 (ON)
LOCk	NONE
ROSc:TRACking	STABLE
USBTMC:PRODUCTID:DECimal?	0
USBTMC:PRODUCTID:HEXadecimal?	0x0000
USBTMC:VENDORID:DECimal?	0
USBTMC:VENDORID:HEXadecimal?	0x0000
VERBose	1 (ON)

Table C-18: Save and recall default settings

Command	Default values
RECAll:SETUp:DESKew	OFF
SAVe:MARKS	”
SAVe:WAVEform:FILEFormat	INTERNAL
SAVe:WAVEform:FORCESAMEFilesize	0

Table C-19: Save on default settings

Command	Default values
SAVEON:COUNt?	1
SAVEON:FILE:AUTOInc	1
SAVEON:FILE:COUNt	1
SAVEON:FILE:TYPE	AUTO
SAVEON:IMAGe	1
SAVEON:LIMit	0
SAVEON:MASK	0
SAVEON:MEASUrement	0
SAVEON:NUMEvents	10
SAVEON:SETUP	0
SAVEON:TRIGger	0
SAVEON:WAVEform	0

Table C-20: Search and mark default values

Command	Default values
SEARCH:MARKALLevents	0
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:CONDition	SOF
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:VALue	XXXXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:FRAMEtype	DATA
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:FORMat	BINARY
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:SIZe	1
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:QUALifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:DIRection	DONTCare
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:FORMat	BINary
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:MODe	STandard
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:VALue	XXXXXXXXX01
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CONDition	SOF
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat	BINary
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue	XXXXXX

Table C-20: Search and mark default values (cont.)

Command	Default values
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:FORMat	BINary
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet	-1
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:SIZe	1
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:VALue	XXXXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:EOFTYPE	ALL
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:ERRTYPE	CRCHEADer
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:FRAMEType	NORMal
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC	XXXXXXXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount	XXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID	XXXXXXXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits	XXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth	XXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat	BINary
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue	XXXXXXXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:CONDition	SYNC
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:FORMat	BINary
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:QUALifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:SIZe	1
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:VALue	XXXXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:ERRTYPE	SYNC
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:FORMat	BINary

Table C-20: Search and mark default values (cont.)

Command	Default values
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:VALue	XXXXXX
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:QUAlifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue	DONTCare
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:QUAlifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue	DONTCare
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier	EQUal
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue	DONTCare
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR	COM
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus	0011111010
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS	1100000101
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CONDition	PATtern
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:DISParity	EITher
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:CHAR	ON
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:DISParity	ON
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:ORDERedset	ON
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:FORMat	ORDERedset
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:LONGORDERedset	TS1
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>	COM SKP SKP SKP
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset	SKP
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>	0011111010 1100001011 1100001011 1100001011 for each of the 4 index elements
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>	1100000101 0011110100 0011110100 0011110100 for each of the 4 index elements
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOBitrate	AUTO

Table C-20: Search and mark default values (cont.)

Command	Default values
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel	AUTO
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:BITRate	9.9000E+36
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:DATASource	CH1
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:HIGH	9.9000E+36
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:LOW	9.9000E+36
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:MID	9.9000E+36
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:HIGH	9.9000E+36
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:LOW	9.9000E+36
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:MID	9.9000E+36
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STANdard	DDR3
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STROBEsource	CH2

Table C-21: Trigger default values

Command	Default values
COUnter:DURation	60 seconds
COUnter:GAIn	1
COUnter:LOGNumber	100 acquisitions
COUnter:LOGTable	ON
	OFF
COUnter:SHOWLog	ON
COUnter:SHOWMeasurement	ON
COUnter:STAte	OFF
COUnter:TYPe	Frequency
COUnter:UPDate	AUTO
TRIGger:A:BUS:CAN:CONDition	SOF
TRIGger:A:BUS:CAN:DATa:DIRection	DONTCare
TRIGger:A:BUS:CAN:DATa:FORMat	BINARY
TRIGger:A:BUS:CAN:DATa:QUALifier	EQUal
TRIGger:A:BUS:CAN:DATa:SIZe	1

Table C-21: Trigger default values (cont.)

Command	Default values
TRIGger:A:BUS:CAN:DATa:VALue	XXXXXXXX
TRIGger:A:BUS:CAN:FRAMEtype	DATA
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:DIRection	DONTCare
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:FORMat	BINARY
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:MODe	STandard
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:VALue	XXXXXXXX01
TRIGger:A:BUS:FLEXRAY:CONDition	SOF
TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat	BINary
TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier	EQUal
TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue	XXXXXX
TRIGger:A:BUS:FLEXRAY:DATa:FORMat	BINary
TRIGger:A:BUS:FLEXRAY:DATa:OFFSet	-1
TRIGger:A:BUS:FLEXRAY:DATa:QUALifier	EQUal
TRIGger:A:BUS:FLEXRAY:DATa:SIZe	1
TRIGger:A:BUS:FLEXRAY:DATa:VALue	XXXXXXXX
TRIGger:A:BUS:FLEXRAY:EOFTYPE	ALL
TRIGger:A:BUS:FLEXRAY:ERRTYPE	CRCHEADer
TRIGger:A:BUS:FLEXRAY:FRAMEType	NORMal
TRIGger:A:BUS:FLEXRAY:HEADER:CRC	XXXXXXXXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount	XXXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID	XXXXXXXXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:INDBits	XXXXX
TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth	XXXXXX
TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat	BINary
TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier	EQUal
TRIGger:A:BUS:FLEXRAY:IDentifier:VALue	XXXXXXXXXX
TRIGger:A:BUS:LIN:CONDition	SYNC
TRIGger:A:BUS:LIN:DATa:FORMat	BINary
TRIGger:A:BUS:LIN:DATa:QUALifier	EQUal
TRIGger:A:BUS:LIN:DATa:SIZe	1
TRIGger:A:BUS:LIN:DATa:VALue	XXXXXXXX
TRIGger:A:BUS:LIN:ERRTYPE	SYNC
TRIGger:A:BUS:LIN:IDentifier:FORMat	BINary
TRIGger:A:BUS:LIN:IDentifier:VALue	XXXXXX
TRIGger:A:BUS:PCIE:CHARacter:CHAR	COM
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus	0011111010
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS	1100000101

Table C-21: Trigger default values (cont.)

Command	Default values
TRIGger:A:BUS:PCIE:CONDition	PATtern
TRIGger:A:BUS:PCIE:DISParity	EITher
TRIGger:A:BUS:PCIE:ERROR	CHARacter
TRIGger:A:BUS:PCIE:FORMat	ORDERedset
TRIGger:A:BUS:PCIE:PATtern:CHAR<x>	COM SKP SKP SKP for each of the 4 index elements
TRIGger:A:BUS:PCIE:PATtern:ORDERedset	SKP
TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>	0011111010 1100001011 1100001011 1100001011 for each of the 4 index elements
TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>	1100000101 0011110100 0011110100 0011110100 for each of the 4 index elements
TRIGger:A:BUS:S8B10B:CHARacter:CHAR	"K28.5"
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus	"0011111010"
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS	"1100000101"
TRIGger:A:BUS:S8B10B:CONDition	PATTERN
TRIGger:A:BUS:S8B10B:DISParity	EITHER
TRIGger:A:BUS:S8B10B:ERROR	CHARACTER
TRIGger:A:BUS:S8B10B:FORMat	CHAR
TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>	CHAR 1 = "K28.5" CHAR 2 = "K28.5"
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>	MINUS1 = "0011111010" MINUS2 = "0011111010"
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>	PLUS1 = "1100000101" PLUS2 = "1100000101"
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:PULSEForm	AMI = PLUSONE CMI = PLUSONE
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:HIGH	AMI = 0.0000
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:LOW	AMI = 0.0000
TRIGger:A:COMMunication:BITRate	1544000
TRIGger:A:COMMunication:CLOCk:POLarity	RISE
TRIGger:A:COMMunication:CODe	NRZ
TRIGger:A:COMMunication:SOUrce	CH1
TRIGger:A:COMMunication:SOUrce:TYPe	DATA
TRIGger:A:COMMunication:STANdard	CUSTOM
TRIGger:{A|B}:EDGE:COUPling	DC
TRIGger:{A|B}:EDGE:SLOpe	RISE
TRIGger:{A|B}:EDGE:SOUrce	A = CH1 B = CH1

Table C-21: Trigger default values (cont.)

Command	Default values
TRIGger:A:HOLDoff:BY	DEFAULT
TRIGger:A:HOLDoff:TIMe	1.5000E-6
TRIGger:{A|B}:LEVel	0.0000
TRIGger:{A|B}:LOGIc:CLAss	PATTERN
TRIGger:{A|B}:LOGIc:FUNCtion	AND
TRIGger:{A|B}:LOGIc:INPut:CH<x>	CH1 = HIGH CH2 = X CH3 = X
TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>	CH4 = X
TRIGger:{A|B}:LOGIc:PATtern:WHEn	TRUE
TRIGger:{A|B}:LOGIc:PATtern:WHEn:LESSLimit	5.0000E-12
TRIGger:{A|B}:LOGIc:PATtern:WHEn:MORELimit	5.0000E-12
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:EDGE	RISE
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:SOUrce	CH2
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:THReshold	1.2000
TRIGger:{A|B}:LOGIc:SETHold:DATa:SOUrce	CH1
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold	1.2000
TRIGger:{A|B}:LOGIc:SETHold:HOLDTime	2.0000E-9
TRIGger:{A|B}:LOGIc:SETHold:SETTime	1.0000E-9
TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x>	CH4 = RISE
TRIGger:{A|B}:LOGIc:STATE:WHEn	TRUE
TRIGger:{A|B}:LOGIc:THReshold:CH<x>	CH4 = 1.2000
TRIGger:A:MODe	AUTO
TRIGger:{A|B}:PULse:CLAss	GLITCH
TRIGger:{A|B}:PULse:GLItch:POLarity	POSITIVE
TRIGger:{A|B}:PULse:GLItch:TRIGIF	ACCEPT
TRIGger:{A|B}:PULse:GLItch:WIDth	2.0000E-9
TRIGger:{A|B}:PULse:RUNT:POLarity	POSITIVE
TRIGger:{A|B}:PULse:RUNT:THReshold:HIGH	1.2000
TRIGger:{A|B}:PULse:RUNT:THReshold:LOW	800.0000E-3
TRIGger:{A|B}:PULse:RUNT:WHEn	OCCURS
TRIGger:{A|B}:PULse:RUNT:WIDth	2.0000E-9
TRIGger:{A|B}:PULse:SOUrce	CH1
TRIGger:{A|B}:PULse:TIMEOut:POLarity	STAYSHIGH
TRIGger:{A|B}:PULse:TIMEOut:TIMe	2.0000E-9
TRIGger:{A|B}:PULse:TRANsition:DELTATime	2.0000E-9
TRIGger:{A|B}:PULse:TRANsition:POLarity	POSITIVE

Table C-21: Trigger default values (cont.)

Command	Default values
TRIGger:{A|B}:PULse:TRANsition:THReshold:HIGH	1.2000
TRIGger:{A|B}:PULse:TRANsition:THReshold:LOW	800.0000E-3
TRIGger:{A|B}:PULse:TRANsition:WHEn	FASTERTHAN
TRIGger:{A|B}:PULse:WIDth:HIGHLimit	2.0000E-9
TRIGger:{A|B}:PULse:WIDth:LOWLimit	2.0000E-9
TRIGger:{A|B}:PULse:WIDth:POLarity	POSITIVE
TRIGger:{A|B}:PULse:WIDth:WHEn	WITHIN
TRIGger:{A|B}:PULse:WINdow:THReshold:HIGH	1.2000
TRIGger:{A|B}:PULse:WINdow:THReshold:LOW	800.0000E-3
TRIGger:{A|B}:PULse:WINdow:WIDTH	2.0000E-9
TRIGger:{A|B}:PULse:WINdow:POLarity	EITHER
TRIGger:{A|B}:PULse:WINdow:POLarity:CH<x>	EITHER
TRIGger:{A|B}:TYPe	EDGE
TRIGger:A:VIDeo:CUSTom:FORMat	INTERLACED
TRIGger:A:VIDeo:FIELD	ALLFIELDS
TRIGger:A:VIDeo:HOLdoff:FIELD	0.0000
TRIGger:A:VIDeo:LINE	1
TRIGger:A:VIDeo:POLarity	NORMAL
TRIGger:A:VIDeo:SCAN	RATE1
TRIGger:A:VIDeo:SOUrce	CH1
TRIGger:A:VIDeo:STANdard	NTSC
TRIGger:B:BY	EVENTS
TRIGger:B:EVENTS:COUNt	2
TRIGger:B:STATE	0
TRIGger:B:TIMe	16.0000E-9
Visual Trigger Commands	
TRIGger:EQUation<1..1>	”
TRIGger:SHOWEQuation	1
VISual:AREA<x>:DISplay	1
VISual:ENAble	0
VISual:AREA<x>:HEIGHT	0.0
VISual:AREA<x>:LABel	”
VISual:AREA<x>:OPERATION<1..1>	IN
VISual:AREA<x>:ROTAtion	0.0
VISual:AREA<x>:SHOWLOGic	1
VISual:AREA<x>:SOURCE<1..1>	CH1
VISual:AREA<x>:VERTICES	0.0, 0.0

Table C-21: Trigger default values (cont.)

Command	Default values
VISual:AREA<x>:WIDTH	0.0
VISual:AREA<x>:XPOSition	0.0
VISual:AREA<x>:YPOSition	0.0
VISual:AREACOLOr	DEFAULT
VISual:ASPECTratio	OFF

Table C-22: Vertical default values

Command	Default values	
CH<x>:BANdwidth	500.0000E+6	
CH<x>:COUPling	DC	
CH<x>:DESKew	0.0000	
CH<x>:INVert	0	
CH<x>:LABel:NAMe	“”	
CH<x>:LABel:XPOS	CH1 = 1.0000	
CH<x>:LABel:YPOS	CH1 = 2.2000	CH3 = 1.4000
	CH2 = 1.8000	CH4 = 1.000
CH<x>:OFFSet	0.0000	
CH<x>:POSition	0.0000	
CH<x>:PROBECOntrol	AUTO	
CH<x>:PROBEFunc:EXTAtten	1.0000	
CH<x>:PROBEFunc:EXTUnits	“V”	
CH<x>:SCAle	100.0000E-3	
CH<x>:TERmination	1.0000E+6	
REF<x>:LABel:NAMe	“”	
REF<x>:LABel:XPOS	1.0000	
REF<x>:LABel:YPOS	–1.0000	
SELect:<wfm>	CH1 = 1	CH3 = 0
	CH2 = 0	CH4 = 0
SELect:CONTROl	CH1	

Table C-23: Waveform default values

Command	Default values
NOTE. *These are not set to factory values when a default setup is executed.* `:DATa INIT` *is used to reset these values.*	
DATa:DESTination	REF1
DATa:ENCdg	RIBINARY
DATa:SOUrce	CH1

Table C-23: Waveform default values (cont.)

Command	Default values
DATa:STARt	1
DATa:STOP	1000

Table C-24: Zoom default values

Command	Default values	
ZOOm:GRAticule:SIZE	80	
ZOOm:MODe	0	
ZOOm:SCROLL:DIREction	STOP	
ZOOm:SCROLL:LOCk	0	
ZOOm:SCROLL:SPEED	1	
ZOOm:ZOOM<x>:<wfm>:DISplay	0	
ZOOm:ZOOM<x>:<wfm>:HORizontal:POSition	Zoom1	50.0000
	Zoom2	30.0000
	Zoom3	70.0000
	Zoom4	90.0000
ZOOm:ZOOM<x>:<wfm>:HORizontal:SCAle	5	
ZOOm:ZOOM<x>:<wfm>:VERTical:POSition	0.0000	
ZOOm:ZOOM<x>:<wfm>:VERTical:SCAle	1.0000	
ZOOm:ZOOM<x>:SCROLLLock	1	
ZOOm:STATE	1	

Appendix D: GPIB interface specifications

This topic describes details of the GPIB remote interface of the instrument. Normally, you will not need this information to use the instrument, but this information can be useful if you are connecting to controllers with unusual configurations.

Interface messages

The following table shows the standard interface messages that the instrument supports.

Table D-1: Standard interface messages

Message	Interface
DCL	Yes
GET	Yes
GTL	Yes
LLO	Yes
PPC	No
PPD	No
PPE	No
PPU	No
SDC	Yes
SPD	Yes
SPE	Yes
TCT	No
UNL	Yes
UNT	Yes
Listen Addresses	Yes
Talk Addresses	Yes

GPIB functions

The following table lists the interface functions and electrical functions subsets supported by this instrument with brief descriptions of each.

Table D-2: Interface functions

Interface Function	Subset	Meaning
Acceptor Handshake	AH1	The instrument can receive multiline messages across the interface from other devices.
Controller	C1	Controller capability, the instrument can control other devices.
Device Clear	DC1	The instrument can respond to both the DCL (Device Clear) interface message and to the Selected Device Clear (SDC) interface message when the instrument is listen-addressed.
Device Trigger	DT1	Device Trigger capability, the instrument does respond to the GET (Group Execute Trigger) interface message.
Electrical	E2	The instrument uses tri-state buffers, which are optimal for high-speed data transfer.
Listener	L4	The instrument becomes a listener when it detects the listen address being sent over the bus with the ATN line asserted. The instrument ceases to be a listener and becomes a talker when it detects the talk address being sent over the bus with the ATN line asserted.
Parallel Poll	PP0	No Parallel Poll Capability, the instrument does not respond to PPC (Parallel Poll Configure), PPO (Parallel POll Disable, PPE (Parallel Poll Enable), or PPU (Parallel Poll Unconfigure) interface messages, nor does it send a status message when the ATN and EOI lines are asserted simultaneously.
Remote/Local	RL1	The instrument can respond to both GTL (Go To Local) and LLO (Local Lock Out) interface messages.
Service Request	SR1	The instrument can assert the SRQ line to notify the controller in charge that it requires service.
Source Handshake	SH1	The instrument can initiate multiline messages to send across the interface to other devices.
Talker	T5	The instrument becomes a talker when it detects the talk address being sent over the bus with the ATN line asserted. The instrument ceases to be a talker and becomes a listener when it detects the listen address being sent over the bus with the ATN line asserted. The instrument also ceases to be a talker when it detects the talk address of another device being sent over the data lines with ATN asserted.

Glossary

ASCII
Acronym for the American Standard Code for Information Interchange. Controllers transmit commands to the instrument using ASCII character encoding.

Address
A 7-bit code that identifies an instrument on the communication bus. The digitizing instrument must have a unique address for the controller to recognize and transmit commands to it.

Backus-Naur Form (BNF)
A standard notation system for command syntax diagrams. The syntax diagrams in this manual use BNF notation.

Controller
A computer or other device that sends commands to and accepts responses from the digitizing instrument.

EOI
A mnemonic referring to the control line End or Identify on the GPIB interface bus. One of the two possible end-of-message terminators.

EOM
A generic acronym referring to the end-of-message terminator. The end-of-message terminator can be either an EOI or the ASCII code for line feed (LF).

Equivalent-Time sampling (ET)
A sampling mode in which the instrument acquires signals over many repetitions of the event. This instrument uses a type of equivalent time sampling called random equivalent time sampling. It utilizes an internal clock that runs asynchronously with respect to the input signal and the signal trigger. The instrument takes samples continuously, independent of the trigger position, and displays them based on the time difference between the sample and the trigger. Although the samples are taken sequentially in time, they are random with respect to the trigger.

Real-Time sampling
A sampling mode where the instrument samples fast enough to completely fill a waveform record from a single trigger event. Use real-time sampling to capture single-shot or transient events.

GPIB
An acronym for General Purpose Interface Bus, the common name for the interface system defined in IEEE Std-488.

IEEE
An acronym for the Institute for Electrical and Electronic Engineers.

TEKSecure
A Tektronix custom command that initializes both waveform and setup memories. This overwrites any previously stored data.

Serial poll
A device on the GPIB bus can request service from the GPIB Controller by asserting the GPIB SRQ line (a Hardware line that is only present on the GPIB communications bus). When a controller acknowledges the SRQ, it "serial polls" each open device on the bus to determine which device on the bus requested service. Any device requesting service returns a status byte with bit 6 set and then unasserts the SRQ line. Devices not requiring service return a status byte with bit 6 cleared.

Index

A

ACQuire:ENHANCEDEnob, 2-105
ACQuire:ENHANCEDEnob:STATE?, 2-105
ACQuire:INTERPEightbit, 2-105
ACQuire:MAGnivu, 2-106
ACQuire:MODe, 2-107
ACQuire:MODe:ACTUal?, 2-108
ACQuire:NUMACq?, 2-109
ACQuire:NUMAVg, 2-110
ACQuire:NUMEnv, 2-110
ACQuire:NUMFRAMESACQuired?, 2-111
ACQuire:NUMSAMples, 2-111
ACQuire:SAMPlingmode, 2-112
ACQuire:STATE, 2-113
ACQuire:STOPAfter, 2-114
ALIas, 2-114
ALIas:CATalog?, 2-115
ALIas:DEFine, 2-115
ALIas:DELEte, 2-116
ALIas:DELEte:ALL, 2-116
ALIas:DELEte:NAMe, 2-117
ALIas:STATE, 2-117
ALLEv?, 2-118
ALLOcate:WAVEform:REF<x>?, 2-118
APPLication:ACTivate, 2-118
APPLication:SCOPEAPP:WINDOW, 2-119
AUTOSet, 2-120
AUXIn:BANdwidth, 2-120
AUXIn:COUPling, 2-121
AUXIn:OFFSet, 2-121
AUXIn:PRObe:AUTOZero, 2-122
AUXIn:PRObe:DEGAUSS, 2-122
AUXIn:PRObe:DEGAUSS:STATE?, 2-123
AUXIn:PRObe:FORCEDRange, 2-123
AUXIn:PRObe:GAIN?, 2-124
AUXIn:PRObe:ID:SERnumber?, 2-124
AUXIn:PRObe:ID:TYPe?, 2-124
AUXIn:PRObe:INPUTMode, 2-125
AUXIn:PRObe:INPUTMode:AOFFSet, 2-125
AUXIn:PRObe:INPUTMode:BOFFSet, 2-126
AUXIn:PRObe:INPUTMode:CMOFFSet, 2-126
AUXIn:PRObe:INPUTMode:DMOFFSet, 2-127
AUXIn:PRObe:RANge, 2-127
AUXIn:PRObe:RESistance?, 2-128
AUXIn:PRObe:SET, 2-128
AUXIn:PRObe:SIGnal, 2-129
AUXIn:PRObe:TIPtype, 2-130
AUXIn:PRObe:UNIts?, 2-130
AUXIn:PROBEFunc:EXTAtten, 2-130
AUXIn:PROBEFunc:EXTDBatten, 2-131
AUXIn:PROBEFunc:EXTUnits, 2-132
AUXIn:VTERm:DUAL:A, 2-133
AUXIn:VTERm:DUAL:B, 2-133
AUXout?, 2-134
AUXout:EDGE, 2-134
AUXout:SOUrce, 2-134

B

BELl, 2-135
BUS:B<x>:CAN:BITRate, 2-136
BUS:B<x>:CAN:BITRate:VALue, 2-137
BUS:B<x>:CAN:PRObe, 2-137
BUS:B<x>:CAN:SOUrce, 2-138
BUS:B<x>:DISplay:DECOde:FILe, 2-139
BUS:B<x>:DISplay:DECOde:STAte, 2-139
BUS:B<x>:ETHERnet:PRObe, 2-140
BUS:B<x>:ETHERnet:SOUrce, 2-141
BUS:B<x>:ETHERnet:SOUrce:DMINus, 2-142
BUS:B<x>:ETHERnet:SOUrce:DPLUs, 2-142
BUS:B<x>:ETHERnet:TYPe, 2-143
BUS:B<x>:FLEXRAY{:PROBe|:SIGnal}, 2-145
BUS:B<x>:FLEXRAY:BITRate, 2-144
BUS:B<x>:FLEXRAY:BITRate:VALue, 2-144
BUS:B<x>:FLEXRAY:CHANnel, 2-145
BUS:B<x>:FLEXRAY:SOUrce, 2-146
BUS:B<x>:I2C:CLOCk:SOUrce, 2-146
BUS:B<x>:I2C:DATa:SOUrce, 2-147
BUS:B<x>:I2C:RWINADDR, 2-148
BUS:B<x>:LABel, 2-148
BUS:B<x>:LIN:BITRate, 2-149
BUS:B<x>:LIN:BITRate:VALue, 2-149
BUS:B<x>:LIN:IDFORmat, 2-150
BUS:B<x>:LIN:POLarity, 2-150
BUS:B<x>:LIN:SOUrce, 2-151
BUS:B<x>:LIN:STANDard, 2-151
BUS:B<x>:MIL1553B:POLarity, 2-152

BUS:B<x>:MIL1553B:RESPonsetime:
MAXimum, 2-153
BUS:B<x>:MIL1553B:RESPonsetime:
MINimum, 2-153
BUS:B<x>:MIL1553B:SOUrce, 2-154
BUS:B<x>:MIPICSITWo:CLOCk:SOUrce, 2-155
BUS:B<x>:MIPICSITWo:CLOCk:TYPe, 2-155
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:
DIFFerential, 2-156
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:
DMINUS, 2-157
BUS:B<x>:MIPICSITWo:LANE<x>:SOUrce:
DPLUS, 2-157
BUS:B<x>:MIPICSITWo:LANE<x>:TYPe, 2-158
BUS:B<x>:MIPIDSIOne:CLOCk:SOUrce, 2-159
BUS:B<x>:MIPIDSIOne:CLOCk:TYPe, 2-160
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:
DIFFerential, 2-160
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:
DMINUS, 2-161
BUS:B<x>:MIPIDSIOne:LANE<x>:SOUrce:
DPLUS, 2-162
BUS:B<x>:MIPIDSIOne:LANE<x>:TYPe, 2-163
BUS:B<x>:PARallel:CLOCk:EDGE, 2-163
BUS:B<x>:PARallel:CLOCk:SOUrce, 2-164
BUS:B<x>:PARallel:ISCLOCKED, 2-165
BUS:B<x>:PARallel:SOURCES, 2-165
BUS:B<x>:PCIE:BITRate, 2-166
BUS:B<x>:PCIE:BITRate:VALue, 2-167
BUS:B<x>:PCIE:HYSTeresis, 2-167
BUS:B<x>:PCIE:LANE, 2-168
BUS:B<x>:PCIE:SOUrce, 2-168
BUS:B<x>:POSition, 2-169
BUS:B<x>:RS232C:BITRate, 2-169
BUS:B<x>:RS232C:DATABits, 2-170
BUS:B<x>:RS232C:DELIMiter, 2-170
BUS:B<x>:RS232C:DISplaymode, 2-171
BUS:B<x>:RS232C:PARity, 2-172
BUS:B<x>:RS232C:POLarity, 2-172
BUS:B<x>:RS232C:SOUrce, 2-173
BUS:B<x>:S8B10B:BITRate, 2-173
BUS:B<x>:S8B10B:BITRate:VALue, 2-174
BUS:B<x>:S8B10B:HYSTeresis, 2-175
BUS:B<x>:S8B10B:SOUrce, 2-175
BUS:B<x>:SPI:BITOrder, 2-176
BUS:B<x>:SPI:CLOCk:POLarity, 2-177
BUS:B<x>:SPI:CLOCk:SOUrce, 2-177
BUS:B<x>:SPI:DATa:POLarity, 2-178
BUS:B<x>:SPI:DATa:SIZe, 2-179
BUS:B<x>:SPI:DATa:SOUrce, 2-179
BUS:B<x>:SPI:FRAMING, 2-180
BUS:B<x>:SPI:IDLETime, 2-180
BUS:B<x>:SPI:SELect:POLarity, 2-181
BUS:B<x>:SPI:SELect:SOUrce, 2-181
BUS:B<x>:TYPe, 2-182
BUS:B<x>:USB:BITRate, 2-183
BUS:B<x>:USB:PRObe, 2-184
BUS:B<x>:USB:SOUrce, 2-184
BUS:B<x>:USB:SOUrce:DMINus, 2-185
BUS:B<x>:USB:SOUrce:DPLUs, 2-186
BUS:B1<x>:DISplay:HIERarchical, 2-135
BUS:B1<x>:DISplay:LAYout, 2-136
BUS:B1<x>:USB:HYSTeresis, 2-136
BUS:CH<x>:LOWTHRESHold, 2-186
BUS:CH<x>:THRESHold, 2-187
BUS:MATH<x>:LOWTHRESHold, 2-187
BUS:MATH<x>:THRESHold, 2-188
BUS:REF<x>:THRESHold, 2-189
BUSY?, 2-189

C

*CAL?, 2-190
CALibrate?, 2-190
CALibrate:CALProbe:CH<x>?, 2-191
CALibrate:INTERNal, 2-191
CALibrate:INTERNal:STARt, 2-192
CALibrate:INTERNal:STATus?, 2-192
CALibrate:PRObestate:CH<x>?, 2-193
CALibrate:RESults?, 2-194
CALibrate:RESults:SPC?, 2-194
CH<x>?, 2-195
CH<x>:ATIACTive?, 2-196
CH<x>:AVAILable?, 2-196
CH<x>:BANdwidth, 2-196
CH<x>:BANdwidth:ENHanced, 2-197
CH<x>:BANdwidth:ENHanced:APPLYtoall, 2-198
CH<x>:BANdwidth:ENHanced:FORCe, 2-198
CH<x>:BANdwidth:ENHanced:STATE?, 2-199
CH<x>:COUPling, 2-200
CH<x>:DESKew, 2-200
CH<x>:FASTAcqcapable?, 2-201
CH<x>:FASTFRamecapable?, 2-201
CH<x>:ICAPture:SOUrce, 2-201
CH<x>:ICAPture:STATE, 2-202

CH<x>:INVert, 2-203
CH<x>:LABel:NAMe, 2-203
CH<x>:LABel:XPOS, 2-204
CH<x>:LABel:YPOS, 2-204
CH<x>:OFFSet, 2-205
CH<x>:OPTIcal:RCVR, 2-206
CH<x>:OPTIcal:WLENgth, 2-206
CH<x>:OPTIcal:WLENgth:LIST?, 2-206
CH<x>:POSition, 2-207
CH<x>:PRObe?, 2-208
CH<x>:PRObe:AUTOZero, 2-208
CH<x>:PRObe:DEGAUSS, 2-208
CH<x>:PRObe:DEGAUSS:STATE?, 2-209
CH<x>:PRObe:FORCEDRange, 2-209
CH<x>:PRObe:GAIN?, 2-210
CH<x>:PRObe:ID?, 2-210
CH<x>:PRObe:ID:SERnumber?, 2-211
CH<x>:PRObe:ID:TYPe?, 2-211
CH<x>:PRObe:INPUTMode, 2-212
CH<x>:PRObe:INPUTMode:AOFFSet, 2-212
CH<x>:PRObe:INPUTMode:BOFFSet, 2-213
CH<x>:PRObe:INPUTMode:CMOFFSet, 2-213
CH<x>:PRObe:INPUTMode:DMOFFSet, 2-214
CH<x>:PRObe:RANge, 2-214
CH<x>:PRObe:RESistance?, 2-215
CH<x>:PRObe:SET, 2-216
CH<x>:PRObe:SIGnal, 2-217
CH<x>:PRObe:TIPtype, 2-217
CH<x>:PRObe:UNIts?, 2-218
CH<x>:PROBECal?, 2-218
CH<x>:PROBECOntrol, 2-219
CH<x>:PROBEFunc:EXTAtten, 2-219
CH<x>:PROBEFunc:EXTDBatten, 2-220
CH<x>:PROBEFunc:EXTUnits, 2-221
CH<x>:SCAle, 2-221
CH<x>:TERmination, 2-222
CH<x>:THRESHold, 2-223
CH<x>:VTERm:BIAS, 2-223
CH<x>:VTERm:DUAL:A, 2-224
CH<x>:VTERm:DUAL:B, 2-224
CHANNELMAPping?, 2-195
CLEAR, 2-225
*CLS, 2-225
CMDBatch, 2-226
Command Groups, 2-9
COUnter, 2-226
COUnter:DURation, 2-227
COUnter:GAIn, 2-228
COUnter:LOGAction, 2-229
COUnter:LOGNumber, 2-229
COUnter:LOGTable, 2-230
COUnter:REFerence, 2-231
COUnter:RESULTs?, 2-231
COUnter:RESULTs:AVGmean?, 2-232
COUnter:RESULTs:DEViation?, 2-233
COUnter:RESULTs:MAXimum?, 2-233
COUnter:RESULTs:MINimum?, 2-234
COUnter:RESULTs:NUMber?, 2-234
COUnter:RESULTs:VALue?, 2-235
COUnter:SHOWLog, 2-235
COUnter:SHOWMeasurement, 2-236
COUnter:STAte, 2-236
COUnter:TYPe, 2-238
COUnter:UPDate, 2-239
COUnter:VIEW, 2-240
CQ<x>:THRESHold, 2-240
CURSor?, 2-241
CURSor:FUNCtion, 2-241
CURSor:HBArs?, 2-242
CURSor:HBArs:DELTa?, 2-242
CURSor:HBArs:POSITION<x>, 2-243
CURSor:HBArs:UNIts?, 2-243
CURSor:LINESTyle, 2-244
CURSor:MODe, 2-244
CURSor:SCREEN:STYle, 2-245
CURSor:SCREEN:XPOSITION<x>, 2-245
CURSor:SCREEN:YPOSITION<x>, 2-246
CURSor:SOUrce<x>, 2-246
CURSor:STATE, 2-247
CURSor:VBArs, 2-248
CURSor:VBArs:DELTa?, 2-248
CURSor:VBArs:POS<x>, 2-249
CURSor:VBArs:POSITION<x>, 2-249
CURSor:VBArs:UNIts, 2-250
CURSor:WAVEform, 2-251
CURSor:WAVEform:HDELTA?, 2-251
CURSor:WAVEform:HPOS<x>?, 2-252
CURSor:WAVEform:POSition<x>, 2-252
CURSor:WAVEform:STYle, 2-253
CURSor:WAVEform:UNIts, 2-253
CURSor:WAVEform:VDELTA?, 2-253
CURSor:XY?, 2-254
CURSor:XY:PRODDELta?, 2-254
CURSor:XY:PRODUCT<x>?, 2-255

CURSor:XY:RADIUS<x>?, 2-255
CURSor:XY:RATDELta?, 2-256
CURSor:XY:RATIO<x>?, 2-256
CURSor:XY:RDELta?, 2-257
CURSor:XY:READOUT, 2-257
CURSor:XY:RECTX<x>, 2-258
CURSor:XY:RECTY<x>, 2-258
CURSor:XY:THDELta?, 2-259
CURSor:XY:THETA<x>?, 2-259
CURSor:XY:XDELta?, 2-260
CURSor:XY:YDELta?, 2-260
CURVe, 2-260
CURVENext?, 2-262
CURVEStream, 2-262

D

D<x>:LABEL, 2-313
D<x>:POSition, 2-314
D<x>:PROBE:ID:SERnumber?, 2-315
D<x>:PROBE:ID:TYPe?, 2-315
D<x>:THRESHold, 2-316
DATa, 2-263
DATa:DESTination, 2-264
DATa:ENCdg, 2-265
DATa:FRAMESTARt, 2-267
DATa:FRAMESTOP, 2-267
DATa:SOUrce, 2-268
DATa:STARt, 2-269
DATa:STOP, 2-269
DATa:SYNCSOUrces, 2-270
DATE, 2-271
*DDT, 2-271
DELEte:SETUp, 2-272
DELEte:WAVEform, 2-273
DESE, 2-273
DIAg:CONTROL:HALT, 2-274
DIAg:CONTROL:LOOP, 2-275
DIAg:EXECUTE, 2-275
DIAg:FAILURES:CLEAR, 2-276
DIAg:ITEM?, 2-276
DIAg:ITEM:FAILURES?, 2-277
DIAg:ITEM:NAMe?, 2-277
DIAg:ITEM:RESULT?, 2-278
DIAg:ITEM:SUBITEMS?, 2-278
DIAg:LEVEL, 2-279
DIAg:LOOPS?, 2-280
DIAg:NAMe?, 2-280
DIAg:NAMe:AREA?, 2-280
DIAg:NAMe:SUBSYS?, 2-281
DIAg:NAMe:TEST?, 2-281
DIAg:NUMITEMS?, 2-282
DIAg:RESults?, 2-282
DIAg:RESults:VERBose?, 2-283
DIAg:SELect:ALL, 2-283
DIAg:SELect:AREA, 2-284
DIAg:SELect:LAST, 2-284
DIAg:SELect:SUBSYS, 2-285
DIAg:SELect:TEST, 2-285
DIAg:STATE, 2-286
DIAg:STOP, 2-287
DISplay?, 2-287
DISplay:CLOCk, 2-288
DISplay:COLOr?, 2-289
DISplay:COLOr:MATHCOLOr, 2-289
DISplay:COLOr:PALEtte:IMAGEView, 2-290
DISplay:COLOr:PALEtte:RECORDView, 2-291
DISplay:COLOr:PALEtte:USEr, 2-291
DISplay:COLOr:PALEtte:USEr:CARet, 2-292
DISplay:COLOr:PALEtte:USEr:CH<x>, 2-292
DISplay:COLOr:PALEtte:USEr:GRAticule, 2-293
DISplay:COLOr:PALEtte:USEr:HIStogram, 2-294
DISplay:COLOr:PALEtte:USEr:MASK, 2-294
DISplay:COLOr:PALEtte:USEr:
MASKHighlight, 2-295
DISplay:COLOr:PALEtte:USEr:MATH<x>, 2-295
DISplay:COLOr:PALEtte:USEr:REF<x>, 2-296
DISplay:COLOr:PALEtte:USEr:WAVEform, 2-296
DISplay:COLOr:REFCOLOr, 2-297
DISplay:DESKew, 2-298
DISplay:DIGital:HEIght, 2-298
DISplay:FILTer, 2-299
DISplay:FORMat, 2-299
DISplay:GRAticule, 2-300
DISplay:INTENSITy?, 2-301
DISplay:INTENSITy:BACKLight, 2-302
DISplay:INTENSITy:WAVEform:
IMAGEView, 2-302
DISplay:INTENSITy:WAVEform:
RECORDView, 2-303
DISplay:PERSistence, 2-303
DISplay:PERSistence:RESET, 2-304
DISplay:SCREENText?, 2-304
DISplay:SCREENText:LABel<x>?, 2-305

DISplay:SCREENTExt:LABel<x>:
FONTCOlor, 2-305
DISplay:SCREENTExt:LABel<x>:
FONTNAme, 2-306
DISplay:SCREENTExt:LABel<x>:FONTSIze, 2-306
DISplay:SCREENTExt:LABel<x>:
FONTSTyle, 2-307
DISplay:SCREENTExt:LABel<x>:NAMe, 2-307
DISplay:SCREENTExt:LABel<x>:STATE, 2-308
DISplay:SCREENTExt:LABel<x>:XPOS, 2-308
DISplay:SCREENTExt:LABel<x>:YPOS, 2-309
DISplay:SCREENTExt:STATE, 2-309
DISplay:SHOWREmote, 2-310
DISplay:STYle, 2-311
DISplay:TRIGBar, 2-311
DISplay:TRIGT, 2-312
DISplay:VARpersist, 2-312
DISplay:WAVEform, 2-313

E

EMail, 2-316
EMail:ATTempts, 2-317
EMail:AUTHLogin, 2-317
EMail:AUTHPassword, 2-318
EMail:COUNt?, 2-318
EMail:FROm, 2-318
EMail:HOSTwanted, 2-319
EMail:IMAGe, 2-319
EMail:MASK, 2-320
EMail:MAXSize, 2-320
EMail:MEASUrement, 2-321
EMail:NUMEMails, 2-321
EMail:SMTPPort, 2-322
EMail:SMTPServer, 2-322
EMail:STATUS?, 2-323
EMail:TIMEOut, 2-323
EMail:TO, 2-324
EMail:TRIGger, 2-324
EMail:WAVEform, 2-325
ERRORDetector:ALERT, 2-325
ERRORDetector:ALIGNCHARacter?, 2-326
ERRORDetector:ALIGNCHARacter:MINus?, 2-326
ERRORDetector:ALIGNCHARacter:PLUS?, 2-327
ERRORDetector:ALIGNCHARacter:
SYMBOL, 2-327
ERRORDetector:ALIGNPRIMitive?, 2-328
ERRORDetector:ALIGNPRIMitive:MINUS?, 2-328
ERRORDetector:ALIGNPRIMitive:
MINus<x>?, 2-329
ERRORDetector:ALIGNPRIMitive:PLUS?, 2-329
ERRORDetector:ALIGNPRIMitive:
PLUS<x>?, 2-329
ERRORDetector:ALIGNPRIMitive:STATE, 2-330
ERRORDetector:ALIGNPRIMitive:
SYMBOL<x>, 2-331
ERRORDetector:ALIGNPRIMitive:
SYMBOLS, 2-330
ERRORDetector:BIT:LENgth, 2-332
ERRORDetector:BIT:SYNCPATtern?, 2-332
ERRORDetector:BIT:SYNCPATtern:
ADVanced, 2-333
ERRORDetector:BIT:SYNCPATtern:
BITString, 2-333
ERRORDetector:BIT:SYNCPATtern:
DISParity<x>, 2-334
ERRORDetector:BIT:SYNCPATtern:
MINus<x>?, 2-334
ERRORDetector:BIT:SYNCPATtern:
PLUS<x>?, 2-335
ERRORDetector:BIT:SYNCPATtern:
SYMBOL<x>, 2-336
ERRORDetector:BIT:SYNCPATtern:
SYMBOLS, 2-335
ERRORDetector:BIT:TEST, 2-336
ERRORDetector:BIT:TEST:COUNt?, 2-337
ERRORDetector:BIT:TEST:DURATION?, 2-337
ERRORDetector:BIT:TEST:MAXALIGNS?, 2-337
ERRORDetector:BIT:TEST:RATE?, 2-338
ERRORDetector:BIT:TEST:RESults?, 2-338
ERRORDetector:BIT:TEST:SECOnds?, 2-339
ERRORDetector:BIT:TEST:STATUS?, 2-339
ERRORDetector:BIT:TEST:STATUS:LOCK?, 2-340
ERRORDetector:BIT:TEST:STATUS:
MAX_AP?, 2-340
ERRORDetector:BIT:TEST:STATUS:
SIGNAL?, 2-341
ERRORDetector:BIT:TEST:STATUS:START?, 2-341
ERRORDetector:BIT:TEST:STATUS:SYNC?, 2-342
ERRORDetector:BIT:TEST:TIME?, 2-342
ERRORDetector:BIT:TEST:TIME:DAYS?, 2-342
ERRORDetector:BIT:TEST:TIME:HOURS?, 2-343
ERRORDetector:BIT:TEST:TIME:
MINUTES?, 2-343
ERRORDetector:BIT:TEST:TIME:SECOnds?, 2-344

ERRORDetector:BITRate, 2-344
ERRORDetector:BITRate:VALue, 2-345
ERRORDetector:CHANnel, 2-345
ERRORDetector:CHARacter?, 2-346
ERRORDetector:CHARacter:TEST, 2-346
ERRORDetector:CHARacter:TEST:COUNt?, 2-347
ERRORDetector:CHARacter:TEST:
DISPCOUNT?, 2-347
ERRORDetector:CHARacter:TEST:
DISPRATE?, 2-348
ERRORDetector:CHARacter:TEST:
DURATION?, 2-348
ERRORDetector:CHARacter:TEST:
MAXALIGNS?, 2-348
ERRORDetector:CHARacter:TEST:RATE?, 2-349
ERRORDetector:CHARacter:TEST:RESults?, 2-349
ERRORDetector:CHARacter:TEST:SECOnds?, 2-350
ERRORDetector:CHARacter:TEST:STATUS?, 2-350
ERRORDetector:CHARacter:TEST:STATUS:
LOCK?, 2-351
ERRORDetector:CHARacter:TEST:STATUS:
MAX_AP?, 2-351
ERRORDetector:CHARacter:TEST:STATUS:
SIGNAL?, 2-352
ERRORDetector:CHARacter:TEST:STATUS:
START?, 2-352
ERRORDetector:CHARacter:TEST:TIME?, 2-353
ERRORDetector:CHARacter:TEST:TIME:
DAYS?, 2-353
ERRORDetector:CHARacter:TEST:TIME:
HOURS?, 2-353
ERRORDetector:CHARacter:TEST:TIME:
MINUTES?, 2-354
ERRORDetector:CHARacter:TEST:TIME:
SECOnds?, 2-354
ERRORDetector:DURATION:COUNt, 2-355
ERRORDetector:DURATION:SECOnds, 2-355
ERRORDetector:DURATION:TIME, 2-356
ERRORDetector:DURATION:TIME:DAYS, 2-356
ERRORDetector:DURATION:TIME:HOURS, 2-357
ERRORDetector:DURATION:TIME:
MINUTES, 2-357
ERRORDetector:DURATION:TIME:
SECOnds, 2-358
ERRORDetector:ERRORLIMIT, 2-358
ERRORDetector:FILE:RECAll, 2-359
ERRORDetector:FILE:SAVe, 2-359
ERRORDetector:FONTSIze, 2-360
ERRORDetector:FRAME?, 2-361
ERRORDetector:FRAme:EOF, 2-361
ERRORDetector:FRAme:INITIALCRCVALue, 2-362
ERRORDetector:FRAme:SOF, 2-362
ERRORDetector:FRAme:TEST, 2-363
ERRORDetector:FRAme:TEST:BADCHARS?, 2-363
ERRORDetector:FRAme:TEST:COUNt?, 2-364
ERRORDetector:FRAme:TEST:DISParity?, 2-364
ERRORDetector:FRAme:TEST:DURATION?, 2-364
ERRORDetector:FRAme:TEST:
MAXALIGNS?, 2-365
ERRORDetector:FRAme:TEST:RATE?, 2-365
ERRORDetector:FRAme:TEST:RESults?, 2-365
ERRORDetector:FRAme:TEST:SECOnds?, 2-366
ERRORDetector:FRAme:TEST:STATUS?, 2-367
ERRORDetector:FRAme:TEST:STATUS:
LOCK?, 2-367
ERRORDetector:FRAme:TEST:STATUS:
MAX_AP?, 2-368
ERRORDetector:FRAme:TEST:STATUS:
SIGNAL?, 2-368
ERRORDetector:FRAme:TEST:STATUS:
START?, 2-368
ERRORDetector:FRAme:TEST:TIME?, 2-369
ERRORDetector:FRAme:TEST:TIME:DAYS?, 2-369
ERRORDetector:FRAme:TEST:TIME:
HOURS?, 2-370
ERRORDetector:FRAme:TEST:TIME:
MINUTES?, 2-370
ERRORDetector:FRAme:TEST:TIME:
SECOnds?, 2-370
ERRORDetector:MAXALIGNS, 2-371
ERRORDetector:PATTERNNAME, 2-371
ERRORDetector:PREset, 2-372
ERRORDetector:PREset:APPLY, 2-373
ERRORDetector:SAVEIMAGE, 2-373
ERRORDetector:SAVEWFM, 2-374
ERRORDetector:SCRAMBLED, 2-375
ERRORDetector:SENDEMAIL, 2-375
ERRORDetector:SIGnaltype, 2-376
ERRORDetector:SKIPSETPRIMitive:
MINus<x>?, 2-376
ERRORDetector:SKIPSETPRIMitive:
PLUS<x>?, 2-377
ERRORDetector:SKIPSETPRIMitive:STATE, 2-377

ERRORDetector:SKIPSETPRIMitive:
SYMBOL<x>, 2-378
ERRORDetector:SKIPSETPRIMitive:
SYMBOLS, 2-378
ERRORDetector:SSC, 2-379
ERRORDetector:STANdard, 2-380
ERRORDetector:STATE, 2-380
ERRORDetector:STATus?, 2-381
ERRORDetector:STOPWHEN, 2-381
ERRORDetector:SYMBOL?, 2-382
ERRORDetector:SYMBOL:TEST, 2-382
ERRORDetector:SYMBOL:TEST:
BADCHARS?, 2-383
ERRORDetector:SYMBOL:TEST:
BITCOUNT?, 2-383
ERRORDetector:SYMBOL:TEST:
BITDURATION?, 2-384
ERRORDetector:SYMBOL:TEST:BITRate?, 2-384
ERRORDetector:SYMBOL:TEST:COUNt?, 2-385
ERRORDetector:SYMBOL:TEST:DISParity?, 2-385
ERRORDetector:SYMBOL:TEST:
DURATION?, 2-385
ERRORDetector:SYMBOL:TEST:
MAXALIGNS?, 2-386
ERRORDetector:SYMBOL:TEST:RATE?, 2-386
ERRORDetector:SYMBOL:TEST:RESults?, 2-386
ERRORDetector:SYMBOL:TEST:SECOnds?, 2-387
ERRORDetector:SYMBOL:TEST:STATUS?, 2-388
ERRORDetector:SYMBOL:TEST:STATUS:
LOCK?, 2-388
ERRORDetector:SYMBOL:TEST:STATUS:
MAX_AP?, 2-389
ERRORDetector:SYMBOL:TEST:STATUS:
SIGNAL?, 2-389
ERRORDetector:SYMBOL:TEST:STATUS:
START?, 2-390
ERRORDetector:SYMBOL:TEST:TIME?, 2-390
ERRORDetector:SYMBOL:TEST:TIME:
DAYS?, 2-390
ERRORDetector:SYMBOL:TEST:TIME:
HOURS?, 2-391
ERRORDetector:SYMBOL:TEST:TIME:
MINUTES?, 2-391
ERRORDetector:SYMBOL:TEST:TIME:
SECOnds?, 2-392
ERRORDetector:TIMEformat, 2-392
ERRORDetector:TYPe, 2-393
*ESE, 2-393
*ESR?, 2-394
EVENT?, 2-394
EVMsg?, 2-395
EVQty?, 2-395
EXPort, 2-396
EXPort:FILEName, 2-396
EXPort:FORMat, 2-397
EXPort:PALEtte, 2-397
EXPort:READOuts, 2-398
EXPort:VIEW, 2-398

F

FACtory, 2-399
FASTAcq?, 2-400
FASTAcq:HIACQRATE, 2-400
FASTAcq:STATE, 2-401
FILESystem?, 2-401
FILESystem:COPy, 2-402
FILESystem:CWD, 2-402
FILESystem:DELEte, 2-403
FILESystem:DIR?, 2-404
FILESystem:MKDir, 2-404
FILESystem:PRInt, 2-405
FILESystem:READFile, 2-405
FILESystem:REName, 2-406
FILESystem:RMDir, 2-407
FILESystem:WRITEFile, 2-407
FPANel:PRESS, 2-408

H

HARDCopy, 2-409
HARDCopy:FILEName, 2-409
HARDCopy:LAYout, 2-410
HARDCopy:PALEtte, 2-410
HARDCopy:PORT, 2-411
HARDCopy:READOuts, 2-412
HARDCopy:VIEW, 2-412
HDR, 2-413
HEADer, 2-414
HISto gram?, 2-414
HISto gram:BOX, 2-415
HISto gram:BOXPcnt, 2-416
HISto gram:COUNt, 2-417
HISto gram:DATa?, 2-417
HISto gram:DISplay, 2-418

HIStogram:FUNCtion, 2-419
HIStogram:MODe, 2-419
HIStogram:SIZe, 2-420
HIStogram:SOUrce, 2-421
HIStogram:STATE, 2-421
HORizontal?, 2-422
HORizontal[:MAIn]:DELay:MODe, 2-444
HORizontal[:MAIn]:DELay:POSition, 2-445
HORizontal[:MAIn]:DELay:TIMe, 2-445
HORizontal[:MAIn]:POSition, 2-446
HORizontal:ACQDURATION?, 2-422
HORizontal:ACQLENGTH?, 2-423
HORizontal:DIGital:MAGnivu:POSition?, 2-423
HORizontal:DIGital:RECOrdlength?, 2-424
HORizontal:DIGital:RECOrdlength:
MAGnivu?, 2-424
HORizontal:DIGital:RECOrdlength:MAIn?, 2-425
HORizontal:DIGital:SAMPLERate?, 2-425
HORizontal:DIGital:SAMPLERate:MAGnivu?, 2-426
HORizontal:DIGital:SAMPLERate:MAIn?, 2-426
HORizontal:DIVisions?, 2-427
HORizontal:FASTframe?, 2-427
HORizontal:FASTframe:COUNt, 2-427
HORizontal:FASTframe:MAXFRames?, 2-428
HORizontal:FASTframe:MULtipleframes:
FRAMESTart:<wfm>, 2-428
HORizontal:FASTframe:MULtipleframes:
MODe, 2-429
HORizontal:FASTframe:MULtipleframes:
NUMFRames:<wfm>, 2-430
HORizontal:FASTframe:REF:FRAme, 2-430
HORizontal:FASTframe:REF:SOUrce, 2-431
HORizontal:FASTframe:SELECTED:CH<x>, 2-431
HORizontal:FASTframe:SELECTED:
MATH<x>, 2-432
HORizontal:FASTframe:SELECTED:REF<x>, 2-432
HORizontal:FASTframe:SELECTED:SOUrce, 2-433
HORizontal:FASTframe:SEQuence, 2-433
HORizontal:FASTframe:SINGLEFramemath, 2-434
HORizontal:FASTframe:SIXteenbit, 2-434
HORizontal:FASTframe:STATE, 2-435
HORizontal:FASTframe:SUMFrame, 2-436
HORizontal:FASTframe:TIMEStamp:ALL:
<wfm>?, 2-436
HORizontal:FASTframe:TIMEStamp:BETWeen:
<wfm>?, 2-437
HORizontal:FASTframe:TIMEStamp:DELTa:
<wfm>?, 2-438
HORizontal:FASTframe:TIMEStamp:FRAMe:
<wfm>?, 2-438
HORizontal:FASTframe:TIMEStamp:REF?, 2-439
HORizontal:FASTframe:TIMEStamp:SELECTED:
<wfm>?, 2-439
HORizontal:FASTframe:TRACk, 2-440
HORizontal:FASTframe:XZEro:ALL:CH<x>?, 2-441
HORizontal:FASTframe:XZEro:ALL:
REF<x>?, 2-441
HORizontal:FASTframe:XZEro:FRAme:
CH<x>?, 2-442
HORizontal:FASTframe:XZEro:FRAme:
REF<x>?, 2-442
HORizontal:FASTframe:XZEro:REF?, 2-443
HORizontal:FASTframe:XZEro:SELECTED:
CH<x>?, 2-443
HORizontal:FASTframe:XZEro:SELECTED:
REF<x>?, 2-443
HORizontal:MAIn?, 2-444
HORizontal:MAIn:INTERPRatio?, 2-446
HORizontal:MAIn:UNIts, 2-447
HORizontal:MAIn:UNIts:STRing, 2-447
HORizontal:MODE, 2-448
HORizontal:MODE:AUTO:LIMITrecordlen, 2-449
HORizontal:MODE:RECOrdlength, 2-449
HORizontal:MODE:SAMPLERate, 2-449
HORizontal:MODE:SCAle, 2-450
HORizontal:POSition, 2-450
HORizontal:ROLL, 2-451
HORizontal:TIMEStamp:CH<x>?, 2-452
HORizontal:TIMEStamp:REF<x>?, 2-452

I

ID?, 2-452
*IDN?, 2-453
IDNMultiscope?, 2-453
IDNMultiscope:A?, 2-453
IDNMultiscope:B?, 2-454
IDNMultiscope:C?, 2-454
IDNMultiscope:D?, 2-454

L

LIMit?, 2-455
LIMit:BEEP, 2-455

LIMit:COMpare, 2-456
LIMit:COMpare:CH<x>, 2-456
LIMit:COMpare:MATH<x>, 2-457
LIMit:COMpare:REF<x>, 2-458
LIMit:EMail, 2-458
LIMit:HARDCopy, 2-459
LIMit:HIGHLIGHTHits, 2-460
LIMit:HIGHLIGHTHits:RESet, 2-460
LIMit:LOCk, 2-461
LIMit:LOG, 2-461
LIMit:SAVEWFM, 2-462
LIMit:SAVEWFM:FILEName, 2-463
LIMit:SRQ, 2-463
LIMit:STATE, 2-464
LIMit:STATus?, 2-465
LIMit:STOPOnviolation, 2-465
LIMit:TEMPlate:DESTination, 2-466
LIMit:TEMPlate:SOUrce, 2-466
LIMit:TEMPlate:STORe, 2-467
LIMit:TEMPlate:TOLerance:HORizontal, 2-467
LIMit:TEMPlate:TOLerance:VERTical, 2-468
LINKTRaining:ACQTime, 2-469
LINKTRaining:ARMscope, 2-469
LINKTRaining:DECOde, 2-471
LINKTRaining:EQUalization, 2-471
LINKTRaining:LANE, 2-472
LINKTRaining:MARK, 2-473
LINKTRaining:SETUP, 2-474
LINKTRaining:STANdard, 2-475
LINKTRaining:STAte, 2-476
LINKTRaining:TRIGgeron, 2-477
LOCk, 2-478
*LRN?, 2-479

M

MARK, 2-479
MARK:CREATE, 2-480
MARK:DELEte, 2-480
MARK:FREE?, 2-481
MARK:SELECTED:END?, 2-481
MARK:SELECTED:FOCUS?, 2-482
MARK:SELECTED:LABel, 2-482
MARK:SELECTED:MARKSINCOLumn?, 2-482
MARK:SELECTED:OWNer?, 2-483
MARK:SELECTED:SOUrce?, 2-483
MARK:SELECTED:STARt?, 2-483
MARK:SELECTED:STATE, 2-484
MARK:SELECTED:ZOOm:POSition?, 2-484
MARK:SELECTED:ZOOm:SCAle?, 2-484
MARK:TOTal?, 2-485
MASK?, 2-485
MASK:AUTOAdjust, 2-486
MASK:AUTOAdjust:HDELTA, 2-486
MASK:AUTOAdjust:VDELTA, 2-487
MASK:AUTOSet:AUTOAdjust, 2-487
MASK:AUTOSet:HPOS, 2-488
MASK:AUTOSet:HSCAle, 2-488
MASK:AUTOSet:MODe, 2-489
MASK:AUTOSet:OFFSETAdj, 2-489
MASK:AUTOSet:STANdard, 2-490
MASK:AUTOSet:TRIGger, 2-496
MASK:AUTOSet:USER:ONE, 2-496
MASK:AUTOSet:USER:TYPe, 2-497
MASK:AUTOSet:USER:ZERo, 2-498
MASK:AUTOSet:VPOS, 2-498
MASK:AUTOSet:VSCAle, 2-499
MASK:COPy:USER, 2-499
MASK:COUNt, 2-500
MASK:COUNt:FAILURES?, 2-500
MASK:COUNt:HITS?, 2-500
MASK:COUNt:SEG<m>:HITS?, 2-501
MASK:COUNt:STATE, 2-501
MASK:COUNt:TESTS?, 2-502
MASK:COUNt:TOTal?, 2-502
MASK:COUNt:VIOLATIONS?, 2-503
MASK:COUNt:WAVEFORMS?, 2-503
MASK:DISplay, 2-504
MASK:FILTer, 2-504
MASK:HIGHLIGHTHits, 2-505
MASK:INVert, 2-505
MASK:LOCk, 2-506
MASK:MARgin:PERCent, 2-506
MASK:MARgin:STATE, 2-507
MASK:MASKPRE:AMPlitude, 2-507
MASK:MASKPRE:HSCAle, 2-508
MASK:MASKPRE:HTRIGPOS, 2-509
MASK:MASKPRE:PATTERNBITS, 2-509
MASK:MASKPRE:PRESAMPBITS, 2-510
MASK:MASKPRE:RECOrdlength, 2-510
MASK:MASKPRE:TRIGTOSAMP, 2-511
MASK:MASKPRE:VOFFSet, 2-511
MASK:MASKPRE:VPOS, 2-512
MASK:MASKPRE:VSCAle, 2-512
MASK:MASKPRE:WIDth, 2-513

MASK:POLarity, 2-514
MASK:SEG<m>, 2-514
MASK:SEG<m>:NR_Pt?, 2-515
MASK:SEG<m>:POINTS, 2-515
MASK:SOUrce, 2-516
MASK:STANdard, 2-516
MASK:STOPOnviolation, 2-523
MASK:TESt:AUX:COMPLetion, 2-524
MASK:TESt:AUX:FAILure, 2-524
MASK:TESt:BEEP:COMPLetion, 2-525
MASK:TESt:BEEP:FAILure, 2-525
MASK:TESt:DELay, 2-526
MASK:TESt:HARDCopy, 2-526
MASK:TESt:LOG:FAILure, 2-527
MASK:TESt:REPeat, 2-528
MASK:TESt:SAMple, 2-528
MASK:TESt:SAMple:THReshold, 2-529
MASK:TESt:SAVEWFM, 2-529
MASK:TESt:SAVEWFM:FILEName, 2-530
MASK:TESt:SRQ:COMPLetion, 2-531
MASK:TESt:SRQ:FAILure, 2-531
MASK:TESt:STATE, 2-532
MASK:TESt:STATUS?, 2-532
MASK:TESt:STOP:FAILure, 2-533
MASK:TESt:THReshold, 2-533
MASK:TESt:WAVEform, 2-534
MASK:USER:AMPlitude, 2-534
MASK:USER:BITRate, 2-535
MASK:USER:HSCAle, 2-535
MASK:USER:HTRIGPOS, 2-536
MASK:USER:LABel, 2-537
MASK:USER:PATTERNBITS, 2-537
MASK:USER:PRESAMPBITS, 2-538
MASK:USER:RECOrdlength, 2-538
MASK:USER:SEG<m>, 2-539
MASK:USER:SEG<m>:NR_Pt?, 2-539
MASK:USER:SEG<m>:POINTS, 2-539
MASK:USER:TRIGTOSAMP, 2-540
MASK:USER:VOFFSet, 2-541
MASK:USER:VPOS, 2-541
MASK:USER:VSCAle, 2-542
MASK:USER:WIDth, 2-542
MATH<x>?, 2-545
MATH<x>:DEFine, 2-545
MATH<x>:FILTer:MODe, 2-546
MATH<x>:FILTer:RISetime, 2-547
MATH<x>:LABel:NAMe, 2-547
MATH<x>:LABel:XPOS, 2-548
MATH<x>:LABel:YPOS, 2-548
MATH<x>:NUMAVg, 2-549
MATH<x>:SPECTral?, 2-550
MATH<x>:SPECTral:CENTER, 2-550
MATH<x>:SPECTral:GATEPOS, 2-551
MATH<x>:SPECTral:GATEWIDTH, 2-551
MATH<x>:SPECTral:LOCk, 2-552
MATH<x>:SPECTral:MAG, 2-553
MATH<x>:SPECTral:PHASE, 2-554
MATH<x>:SPECTral:REFLevel, 2-554
MATH<x>:SPECTral:REFLEVELOffset, 2-555
MATH<x>:SPECTral:RESBw, 2-556
MATH<x>:SPECTral:SPAN, 2-556
MATH<x>:SPECTral:SUPPress, 2-557
MATH<x>:SPECTral:UNWRap, 2-558
MATH<x>:SPECTral:WINdow, 2-558
MATH<x>:THRESHold, 2-559
MATH<x>:UNITString, 2-560
MATH<x>:VERTical:AUTOSCale, 2-560
MATH<x>:VERTical:POSition, 2-561
MATH<x>:VERTical:SCAle, 2-562
MATHArbflt<x>:FILepath, 2-543
MATHArbflt<x>:READFile, 2-543
MATHVAR?, 2-544
MATHVAR:VAR<x>, 2-544
MEASUrement?, 2-563
MEASUrement:ANNOTation:IMMEDSTAte, 2-564
MEASUrement:ANNOTation:NUMX?, 2-564
MEASUrement:ANNOTation:NUMY?, 2-565
MEASUrement:ANNOTation:STATE, 2-565
MEASUrement:ANNOTation:TYPe, 2-566
MEASUrement:ANNOTation:X<x>?, 2-566
MEASUrement:ANNOTation:Y<x>?, 2-566
MEASUrement:GATing, 2-567
MEASUrement:IMMed?, 2-568
MEASUrement:IMMed:DELay?, 2-568
MEASUrement:IMMed:DELay:DIREction, 2-568
MEASUrement:IMMed:DELay:EDGE[1], 2-569
MEASUrement:IMMed:DELay:EDGE2, 2-570
MEASUrement:IMMed:METHod, 2-571
MEASUrement:IMMed:NOISe, 2-571
MEASUrement:IMMed:REFLevel?, 2-572
MEASUrement:IMMed:REFLevel:ABSolute:
HIGH, 2-572
MEASUrement:IMMed:REFLevel:ABSolute:
LOW, 2-573

MEASUrement:IMMed:REFLevel:ABSolute:MID<x>, 2-573
MEASUrement:IMMed:REFLevel:METHod, 2-574
MEASUrement:IMMed:REFLevel:PERCent:HIGH, 2-575
MEASUrement:IMMed:REFLevel:PERCent:LOW, 2-575
MEASUrement:IMMed:REFLevel:PERCent:MID<x>, 2-576
MEASUrement:IMMed:SOUrce<x>, 2-577
MEASUrement:IMMed:SOUrce<x>:SIGType, 2-578
MEASUrement:IMMed:TYPe, 2-578
MEASUrement:IMMed:UNIts?, 2-583
MEASUrement:IMMed:VALue?, 2-583
MEASUrement:MEAS<x>?, 2-584
MEASUrement:MEAS<x>:COUNt?, 2-584
MEASUrement:MEAS<x>:DELay?, 2-585
MEASUrement:MEAS<x>:DELay:DIREction, 2-585
MEASUrement:MEAS<x>:DELay:EDGE<x>, 2-586
MEASUrement:MEAS<x>:MAXimum?, 2-586
MEASUrement:MEAS<x>:MEAN?, 2-587
MEASUrement:MEAS<x>:METHod, 2-587
MEASUrement:MEAS<x>:MINImum?, 2-588
MEASUrement:MEAS<x>:NOISe, 2-588
MEASUrement:MEAS<x>:REFLevel?, 2-589
MEASUrement:MEAS<x>:REFLevel:ABSolute:HIGH, 2-589
MEASUrement:MEAS<x>:REFLevel:ABSolute:LOW, 2-590
MEASUrement:MEAS<x>:REFLevel:ABSolute:MID<x>, 2-590
MEASUrement:MEAS<x>:REFLevel:METHod, 2-591
MEASUrement:MEAS<x>:REFLevel:PERCent:HIGH, 2-592
MEASUrement:MEAS<x>:REFLevel:PERCent:LOW, 2-592
MEASUrement:MEAS<x>:REFLevel:PERCent:MID<x>, 2-593
MEASUrement:MEAS<x>:SOUrce<x>, 2-594
MEASUrement:MEAS<x>:SOUrce<x>:SIGType, 2-594
MEASUrement:MEAS<x>:STATE, 2-595
MEASUrement:MEAS<x>:STDdev?, 2-596
MEASUrement:MEAS<x>:TYPe, 2-596
MEASUrement:MEAS<x>:UNIts?, 2-601
MEASUrement:MEAS<x>:VALue?, 2-601
MEASUrement:METHod, 2-602
MEASUrement:NOISe, 2-602
MEASUrement:REFLevel:ABSolute:HIGH, 2-603
MEASUrement:REFLevel:ABSolute:LOW, 2-604
MEASUrement:REFLevel:ABSolute:MID<x>, 2-605
MEASUrement:REFLevel:METHod, 2-605
MEASUrement:REFLevel:PERCent:HIGH, 2-606
MEASUrement:REFLevel:PERCent:LOW, 2-607
MEASUrement:REFLevel:PERCent:MID<x>, 2-608
MEASUrement:SOUrce<x>:SIGType, 2-608
MEASUrement:STATIstics:COUNt, 2-609
MEASUrement:STATIstics:MODe, 2-610
MEASUrement:STATIstics:WEIghting, 2-610
MULTiscope:CONFig, 2-611
MULTiscope:EXIT, 2-612
MULTiscope:RESTART, 2-612
MULTiscope:STATUS?, 2-612

N

NEWpass, 2-613

O

*OPC, 2-613
OPCEXtended, 2-614
*OPT?, 2-616

P

PASSWord, 2-616
*PSC, 2-616
*PUD, 2-617

R

*RCL, 2-618
RECAll:MASK, 2-618
RECAll:SETUp, 2-619
RECAll:SETUp:DESKew, 2-620
RECAll:WAVEform, 2-620
REF<x>:HORizontal:POSition, 2-621
REF<x>:LABel?, 2-622
REF<x>:LABel:NAMe, 2-622
REF<x>:LABel:XPOS, 2-623
REF<x>:LABel:YPOS, 2-623
REF<x>:THRESHold, 2-624
REF<x>:VERTical:POSition, 2-624
REF<x>:VERTical:SCAle, 2-625

REM, 2-626
ROSc:OUT:FREQuency, 2-627
ROSc:OUT:ULTRAsync, 2-627
ROSc:SOUrce, 2-628
ROSc:STATE?, 2-628
ROSc:TRACking, 2-629
*RST, 2-630

S

*SAV, 2-630
SAVe:EVENTtable:BUS<x>, 2-631
SAVe:MARKS, 2-631
SAVe:MASK, 2-632
SAVe:SETUp, 2-632
SAVe:WAVEform, 2-633
SAVe:WAVEform:DATa:STARt, 2-634
SAVe:WAVEform:DATa:STOP, 2-635
SAVe:WAVEform:FILEFormat, 2-635
SAVe:WAVEform:FORCESAMEFilesize, 2-637
SAVEON, 2-637
SAVEON:COUNt?, 2-638
SAVEON:FILE:AUTOInc, 2-638
SAVEON:FILE:COUNt, 2-639
SAVEON:FILE:DEST, 2-639
SAVEON:FILE:NAME, 2-640
SAVEON:FILE:TYPE, 2-641
SAVEON:IMAGe, 2-642
SAVEON:LIMit, 2-643
SAVEON:MASK, 2-643
SAVEON:MEASUrement, 2-644
SAVEON:NUMEvents, 2-645
SAVEON:SETUP, 2-645
SAVEON:TRIGger, 2-646
SAVEON:WAVEform, 2-647
*SDS, 2-648
SEARCH:MARKALLevents, 2-648
SEARCH:SEARCH<x>, 2-649
SEARCH:SEARCH<x>:COPy, 2-649
SEARCH:SEARCH<x>:STATE, 2-649
SEARCH:SEARCH<x>:TOTAL?, 2-650
SEARCH:SEARCH<x>:TRIGer:A:BUS:PCIE:SYNCHeader, 2-774
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:CONDition, 2-650
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:FORMat, 2-651
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:QUALifier, 2-651
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:SIZe, 2-652
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:DATa:VALue, 2-652
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:FRAMEtype, 2-653
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:DIRection, 2-653
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:FORMat, 2-654
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:MODe, 2-655
SEARCH:SEARCH<x>:TRIGger:A:BUS:CAN:IDentifier:VALue, 2-655
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:CONDition, 2-656
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:FORMat, 2-657
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:OFFSet, 2-657
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:SIZe, 2-658
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:DATa:VALue, 2-659
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:FORMat, 2-659
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:DESTinationaddr:VALue, 2-660
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:FORMat, 2-661
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:VALue, 2-662
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat, 2-663
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue, 2-663
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat, 2-667
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue, 2-668

SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat, 2-664
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue, 2-665
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat, 2-666
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue, 2-666
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:FORMat, 2-669
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QTAG:VALue, 2-669
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:QUALifier, 2-670
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat, 2-671
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue, 2-671
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat, 2-672
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue, 2-673
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat, 2-674
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue, 2-675
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat, 2-675
SEARCH:SEARCH<x>:TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue, 2-676
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CONDition, 2-677
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat, 2-677
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier, 2-678
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue, 2-679
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:FORMat, 2-679
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:OFFSet, 2-680
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:QUALifier, 2-680
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:SIZe, 2-681
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:DATa:VALue, 2-681
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:EOFTYPE, 2-682
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:ERRTYPE, 2-682
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:FRAMEType, 2-683
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CRC, 2-684
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount, 2-684
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID, 2-685
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:INDBits, 2-685
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth, 2-686
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat, 2-686
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier, 2-687
SEARCH:SEARCH<x>:TRIGger:A:BUS:FLEXRAY:IDentifier:VALue, 2-687
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:FORMat, 2-688
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:MODe, 2-689
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:TYPe, 2-689
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:ADDress:VALue, 2-690
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:CONDition, 2-691
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:DIRection, 2-691
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:FORMat, 2-692

SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:SIZe, 2-693
SEARCH:SEARCH<x>:TRIGger:A:BUS:I2C:DATa:VALue, 2-693
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:CONDition, 2-694
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:FORMat, 2-694
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:QUALifier, 2-695
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:SIZe, 2-695
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:DATa:VALue, 2-696
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:ERRTYPE, 2-696
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:FORMat, 2-697
SEARCH:SEARCH<x>:TRIGger:A:BUS:LIN:IDentifier:VALue, 2-698
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat, 2-698
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier, 2-699
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue, 2-700
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt, 2-700
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat, 2-701
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:PARity, 2-702
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress, 2-703
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:SUBADdress:FORMat, 2-703
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:COMMAND:TRBit, 2-704
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:CONDition, 2-705
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:FORMat, 2-707
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:PARity, 2-707
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:DATA:VALue, 2-708
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:ERRTYPE, 2-709
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:FORMat, 2-709
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:QUALifier, 2-710
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:VALue, 2-711
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BCR, 2-712
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:BUSY, 2-712
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:DBCA, 2-713
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:INSTR, 2-714
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:ME, 2-715
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:PARity, 2-716
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SRQ, 2-716
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:SUBSF, 2-717
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:STATUS:BIT:TF, 2-718
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:LESSLimit, 2-719
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:MORELimit, 2-719
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIL1553B:TIME:QUALifier, 2-720
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:CONDition, 2-721
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTLong, 2-722
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:DTShort, 2-723
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ERRor, 2-724
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:ESCape, 2-724

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:OFFset:VALue, 2-725
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:FORMat, 2-726
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PACKetdata:VALue, 2-726
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:FORMat, 2-727
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:SIZe, 2-728
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PAYLoad:VALue, 2-728
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:PIXel:VALue, 2-729
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:QUALifier, 2-730
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RAW:VALue, 2-730
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:QUALifier, 2-731
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:BLUe:VALue, 2-732
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:QUALifier, 2-732
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:GREen:VALue, 2-733
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:QUALifier, 2-734
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:RGB:RED:VALue, 2-734
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:VC, 2-735
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WARNing, 2-736
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:WC:VALue, 2-736
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:QUALifier, 2-737
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:U:VALue, 2-737
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:QUALifier, 2-738
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:V:VALue, 2-739
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:QUALifier, 2-740
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPICSITWo:YUV:Y:VALue, 2-740
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ACKERRreport, 2-741
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:CONDition, 2-742
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DIRection, 2-743
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTLong, 2-744
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:DTShort, 2-745
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ERRor, 2-746
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:ESCape, 2-747
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:OFFset:VALue, 2-747
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:FORMat, 2-748
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUEOne, 2-749
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PACKetdata:VALUETwo, 2-750
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:FORMat, 2-750
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:SIZe, 2-751
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PAYLoad:VALue, 2-752
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:PIXel:VALue, 2-752
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:QUALifier, 2-753
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:BLUe:VALue, 2-754
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:QUALifier, 2-755
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:GREen:VALue, 2-755
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:QUALifier, 2-756
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:RGB:RED:VALue, 2-757
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:VC, 2-757

SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WARNing, 2-758
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:WC:VALue, 2-759
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:QUAlifier, 2-759
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CB:VALue, 2-760
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:QUAlifier, 2-760
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:CR:VALue, 2-761
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:QUAlifier, 2-762
SEARCH:SEARCH<x>:TRIGger:A:BUS:MIPIDSIOne:YCBCR:Y:VALue, 2-762
SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:FORMat, 2-763
SEARCH:SEARCH<x>:TRIGger:A:BUS:PARallel:DATa:VALue, 2-764
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:CHAR, 2-764
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:MINus, 2-765
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CHARacter:SYMbol:PLUS, 2-766
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:CONDition, 2-766
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:DISParity, 2-767
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:CHAR, 2-768
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:DISParity, 2-768
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:ORDERedset, 2-769
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:ERROR:SYNC, 2-769
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:FORMat, 2-770
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:LONGORDERedset, 2-771
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, 2-771
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:ORDERedset, 2-772
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:MINus<x>, 2-773
SEARCH:SEARCH<x>:TRIGger:A:BUS:PCIE:PATtern:SYMbol:PLUS<x>, 2-774
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:CONDition, 2-775
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:FORMat, 2-775
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:SIZe, 2-776
SEARCH:SEARCH<x>:TRIGger:A:BUS:RS232C:DATa:VALue, 2-777
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:CHAR, 2-777
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:MINus, 2-778
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:PLUS, 2-778
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:CONDition, 2-779
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:DISParity, 2-780
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:CHAR, 2-781
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:ERROR:DISParity, 2-781
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:FORMat, 2-782
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:CHAR<x>, 2-783
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:MINus<x>, 2-783
SEARCH:SEARCH<x>:TRIGger:A:BUS:S8B10B:PATtern:SYMbol:PLUS<x>, 2-784
SEARCH:SEARCH<x>:TRIGger:A:BUS:SOUrce, 2-785
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:CONDition, 2-785
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:FORMat, 2-786
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:SIZe, 2-786
SEARCH:SEARCH<x>:TRIGger:A:BUS:SPI:DATa:VALue, 2-787
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:FORMat, 2-788

SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:HIVALue, 2-788
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ADDress:VALue, 2-789
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:CHAR, 2-790
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:MINus, 2-790
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CHARacter:SYMbol:PLUS, 2-791
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:CONDition, 2-791
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:FORMat, 2-792
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:HIVALue, 2-793
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:OFFSet, 2-793
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:SIZe, 2-794
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:TYPe, 2-794
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DATa:VALue, 2-795
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:DISParity, 2-796
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:FORMat, 2-796
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:HIVALue, 2-797
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ENDPoint:VALue, 2-798
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:CHAR, 2-798
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:DISParity, 2-799
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERROR:ORDERedset, 2-800
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:ERRTYPE, 2-800
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:FORMat, 2-801
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:HANDSHAKEType, 2-801
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:LONGORDERedset, 2-802
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets, 2-803
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:LMPPACKets, 2-803
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PACKets:TPPACKets, 2-804
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:CHAR<x>, 2-805
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:NUMSymbols, 2-805
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:ORDERedset, 2-806
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>, 2-806
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>, 2-807
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:QUALifier, 2-807
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FORMat, 2-808
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SOF:FRAMENUMber, 2-809
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPECIALType, 2-810
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:ET:VALue, 2-810
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:FORMat, 2-811
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:HUB:VALue, 2-812
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:FORMat, 2-812
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:PORT:VALue, 2-813
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SC:VALue, 2-814
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:SPLIT:SE:VALue, 2-814
SEARCH:SEARCH<x>:TRIGger:A:BUS:USB:TOKENType, 2-815
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOBitrate, 2-816
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:AUTOReflevel, 2-816
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:BITRate, 2-817

SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSActive, 2-818
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSLevel, 2-818
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSMode, 2-819
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:CSSource, 2-819
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:DATASource, 2-820
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:HYSteresis, 2-820
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MARgin, 2-821
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MAXCas, 2-821
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:MINCas, 2-822
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:HIGH, 2-822
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:LOW, 2-823
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:DATa:MID, 2-823
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:HIGH, 2-824
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:LOW, 2-825
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:REFLevel:STROBE:MID, 2-825
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STANdard, 2-826
SEARCH:SEARCH<x>:TRIGger:A:DDRMemory:STROBEsource, 2-826
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:CH<x>, 2-827
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:MATH<x>, 2-828
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SLOpe:REF<x>, 2-828
SEARCH:SEARCH<x>:TRIGger:A:EDGE:SOUrce, 2-829
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:CH<x>, 2-829
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:MATH<x>, 2-830
SEARCH:SEARCH<x>:TRIGger:A:GLItch:POLarity:REF<x>, 2-831
SEARCH:SEARCH<x>:TRIGger:A:GLItch:TRIGIF, 2-831
SEARCH:SEARCH<x>:TRIGger:A:GLItch:WIDth, 2-832
SEARCH:SEARCH<x>:TRIGger:A:LEVel:CH<x>, 2-832
SEARCH:SEARCH<x>:TRIGger:A:LEVel:MATH<x>, 2-833
SEARCH:SEARCH<x>:TRIGger:A:LEVel:REF<x>, 2-833
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:FUNCtion, 2-834
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:CH<x>, 2-834
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:MATH<x>, 2-835
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:INPut:REF<x>, 2-836
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:CH<x>, 2-836
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:MATH<x>, 2-837
SEARCH:SEARCH<x>:TRIGger:A:LOGIc:THReshold:REF<x>, 2-837
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn, 2-838
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:LESSLimit, 2-838
SEARCH:SEARCH<x>:TRIGger:A:PATtern:WHEn:MORELimit, 2-839
SEARCH:SEARCH<x>:TRIGger:A:PULse:SOUrce, 2-840
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:CH<x>, 2-840
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:MATH<x>, 2-841
SEARCH:SEARCH<x>:TRIGger:A:RUNT:POLarity:REF<x>, 2-842
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:CH<x>, 2-842
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:MATH<x>, 2-843
SEARCH:SEARCH<x>:TRIGger:A:RUNT:THReshold:HIGH:REF<x>, 2-843

SEARCH:SEARCH<x>:TRIGger:A:RUNT:
THReshold:LOW:CH<x>, 2-844
SEARCH:SEARCH<x>:TRIGger:A:RUNT:
THReshold:LOW:MATH<x>, 2-844
SEARCH:SEARCH<x>:TRIGger:A:RUNT:
THReshold:LOW:REF<x>, 2-845
SEARCH:SEARCH<x>:TRIGger:A:RUNT:
WHEn, 2-845
SEARCH:SEARCH<x>:TRIGger:A:RUNT:
WIDth, 2-846
SEARCH:SEARCH<x>:TRIGger:A:SETHold:
CLOCk:EDGE, 2-846
SEARCH:SEARCH<x>:TRIGger:A:SETHold:
CLOCk:SOUrce, 2-847
SEARCH:SEARCH<x>:TRIGger:A:SETHold:
CLOCk:THReshold:CH<x>, 2-847
SEARCH:SEARCH<x>:TRIGger:A:SETHold:
CLOCk:THReshold:MATH<x>, 2-848
SEARCH:SEARCH<x>:TRIGger:A:SETHold:
CLOCk:THReshold:REF<x>, 2-849
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:
SOUrce, 2-849
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:
THReshold:CH<x>, 2-850
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:
THReshold:MATH<x>, 2-851
SEARCH:SEARCH<x>:TRIGger:A:SETHold:DATa:
THReshold:REF<x>, 2-851
SEARCH:SEARCH<x>:TRIGger:A:SETHold:
HOLDTime, 2-852
SEARCH:SEARCH<x>:TRIGger:A:SETHold:
SETTime, 2-852
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:
EDGE, 2-853
SEARCH:SEARCH<x>:TRIGger:A:STATE:CLOCk:
SOUrce, 2-853
SEARCH:SEARCH<x>:TRIGger:A:STATE:
WHEn, 2-854
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:
POLarity:CH<x>, 2-854
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:
POLarity:MATH<x>, 2-855
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:
POLarity:REF<x>, 2-856
SEARCH:SEARCH<x>:TRIGger:A:TIMEOut:
TIMe, 2-856
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
DELTATime, 2-857
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
POLarity:CH<x>, 2-857
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
POLarity:MATH<x>, 2-858
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
POLarity:REF<x>, 2-858
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
THReshold:HIGH:CH<x>, 2-859
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
THReshold:HIGH:MATH<x>, 2-860
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
THReshold:HIGH:REF<x>, 2-860
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
THReshold:LOW:CH<x>, 2-861
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
THReshold:LOW:MATH<x>, 2-861
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
THReshold:LOW:REF<x>, 2-862
SEARCH:SEARCH<x>:TRIGger:A:TRANsition:
WHEn, 2-862
SEARCH:SEARCH<x>:TRIGger:A:TYPe, 2-863
SEARCH:SEARCH<x>:TRIGger:A:WIDth:
HIGHLimit, 2-864
SEARCH:SEARCH<x>:TRIGger:A:WIDth:
LOWLimit, 2-865
SEARCH:SEARCH<x>:TRIGger:A:WIDth:
POLarity:CH<x>, 2-865
SEARCH:SEARCH<x>:TRIGger:A:WIDth:
POLarity:MATH<x>, 2-866
SEARCH:SEARCH<x>:TRIGger:A:WIDth:
POLarity:REF<x>, 2-866
SEARCH:SEARCH<x>:TRIGger:A:WIDth:
WHEn, 2-867
SEARCH:SEARCH<x>:TRIGger:A:WINdow:
EVENT, 2-867
SEARCH:SEARCH<x>:TRIGger:A:WINdow:
THReshold:HIGH:CH<x>, 2-868
SEARCH:SEARCH<x>:TRIGger:A:WINdow:
THReshold:HIGH:MATH<x>, 2-869
SEARCH:SEARCH<x>:TRIGger:A:WINdow:
THReshold:HIGH:REF<x>, 2-869
SEARCH:SEARCH<x>:TRIGger:A:WINdow:
THReshold:LOW:CH<x>, 2-870
SEARCH:SEARCH<x>:TRIGger:A:WINdow:
THReshold:LOW:MATH<x>, 2-870

SEARCH:SEARCH<x>:TRIGger:A:WINdow:
THReshold:LOW:REF<x>, 2-871
SEARCH:SEARCH<x>:TRIGger:A:WINdow:
WIDth, 2-871
SEARCH:STOP, 2-872
SELect?, 2-872
SELect:<wfm>, 2-876
SELect:B<x>, 2-873
SELect:CH<x>, 2-873
SELect:CONTROl, 2-874
SELect:D<x>, 2-876
SELect:DALL, 2-875
SELect:DIGTraces:COMbination, 2-875
SELect:DIGTraces:LISt, 2-875
SET?, 2-877
SETUp:NAMe, 2-878
*SRE, 2-879
*STB?, 2-879

T

TEKSecure, 2-880
TEST, 2-880
TEST:RESults?, 2-881
TEST:RESults:VERBose?, 2-881
TEST:STOP, 2-882
TIME, 2-882
*TRG, 2-883
TRIGger, 2-883
TRIGger:{A|B}:EDGE?, 2-884
TRIGger:{A|B}:EDGE:COUPling, 2-885
TRIGger:{A|B}:EDGE:COUPling:CH<x>, 2-886
TRIGger:{A|B}:EDGE:SLOpe, 2-886
TRIGger:{A|B}:EDGE:SLOpe:AUX, 2-887
TRIGger:{A|B}:EDGE:SLOpe:CH<x>, 2-888
TRIGger:{A|B}:EDGE:SOUrce, 2-888
TRIGger:{A|B}:LEVel, 2-889
TRIGger:{A|B}:LEVel:CH<x>, 2-890
TRIGger:{A|B}:LOGIc?, 2-890
TRIGger:{A|B}:LOGIc:CLAss, 2-891
TRIGger:{A|B}:LOGIc:FUNCtion, 2-892
TRIGger:{A|B}:LOGIc:INPut?, 2-892
TRIGger:{A|B}:LOGIc:INPut:ALL, 2-893
TRIGger:{A|B}:LOGIc:INPut:CH<x>, 2-893
TRIGger:{A|B}:LOGIc:INPut:FORMat, 2-894
TRIGger:{A|B}:LOGIc:PATtern?, 2-895
TRIGger:{A|B}:LOGIc:PATtern:INPut:CH<x>, 2-895
TRIGger:{A|B}:LOGIc:PATtern:WHEn, 2-896
TRIGger:{A|B}:LOGIc:PATtern:WHEn:
LESSLimit, 2-897
TRIGger:{A|B}:LOGIc:PATtern:WHEn:
MORELimit, 2-898
TRIGger:{A|B}:LOGIc:SETHold?, 2-899
TRIGger:{A|B}:LOGIc:SETHold:CLOCk?, 2-899
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:
EDGE, 2-900
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:
SOUrce, 2-900
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:
THReshold, 2-901
TRIGger:{A|B}:LOGIc:SETHold:CLOCk:
THReshold:CH<x>, 2-902
TRIGger:{A|B}:LOGIc:SETHold:DATa?, 2-903
TRIGger:{A|B}:LOGIc:SETHold:DATa:
SOUrce, 2-903
TRIGger:{A|B}:LOGIc:SETHold:DATa:
THReshold, 2-904
TRIGger:{A|B}:LOGIc:SETHold:DATa:THReshold:
CH<x>, 2-904
TRIGger:{A|B}:LOGIc:SETHold:HOLDTime, 2-905
TRIGger:{A|B}:LOGIc:SETHold:QUALify, 2-906
TRIGger:{A|B}:LOGIc:SETHold:SETTime, 2-907
TRIGger:{A|B}:LOGIc:STATE?, 2-908
TRIGger:{A|B}:LOGIc:STATE:CLOck:
SOUrce, 2-908
TRIGger:{A|B}:LOGIc:STATE:INPut:CH<x>, 2-909
TRIGger:{A|B}:LOGIc:STATE:WHEn, 2-909
TRIGger:{A|B}:LOGIc:THReshold?, 2-910
TRIGger:{A|B}:LOGIc:THReshold:CH<x>, 2-911
TRIGger:{A|B}:LOWerthreshold:CH<x>, 2-911
TRIGger:{A|B}:PULse?, 2-912
TRIGger:{A|B}:PULse:CLAss, 2-913
TRIGger:{A|B}:PULse:GLItch?, 2-914
TRIGger:{A|B}:PULse:GLItch:
LOWPASSfilter, 2-914
TRIGger:{A|B}:PULse:GLItch:POLarity, 2-915
TRIGger:{A|B}:PULse:GLItch:POLarity:
CH<x>, 2-916
TRIGger:{A|B}:PULse:GLItch:QUALify, 2-917
TRIGger:{A|B}:PULse:GLItch:TRIGIF, 2-918
TRIGger:{A|B}:PULse:GLItch:WIDth, 2-918
TRIGger:{A|B}:PULse:RUNT?, 2-919
TRIGger:{A|B}:PULse:RUNT:POLarity, 2-920
TRIGger:{A|B}:PULse:RUNT:POLarity:
CH<x>, 2-920

TRIGger:{A|B}:PULse:RUNT:QUALify, 2-921
TRIGger:{A|B}:PULse:RUNT:THReshold?, 2-922
TRIGger:{A|B}:PULse:RUNT:THReshold:
BOTh, 2-923
TRIGger:{A|B}:PULse:RUNT:THReshold:
HIGH, 2-923
TRIGger:{A|B}:PULse:RUNT:THReshold:
LOW, 2-924
TRIGger:{A|B}:PULse:RUNT:WHEn, 2-925
TRIGger:{A|B}:PULse:RUNT:WIDth, 2-925
TRIGger:{A|B}:PULse:SOUrce, 2-926
TRIGger:{A|B}:PULse:TIMEOut?, 2-927
TRIGger:{A|B}:PULse:TIMEOut:
LOWPASSfilter, 2-927
TRIGger:{A|B}:PULse:TIMEOut:POLarity, 2-928
TRIGger:{A|B}:PULse:TIMEOut:POLarity:
CH<x>, 2-928
TRIGger:{A|B}:PULse:TIMEOut:QUALify, 2-929
TRIGger:{A|B}:PULse:TIMEOut:TIMe, 2-930
TRIGger:{A|B}:PULse:TRANsition?, 2-931
TRIGger:{A|B}:PULse:TRANsition:
DELTATime, 2-931
TRIGger:{A|B}:PULse:TRANsition:POLarity, 2-932
TRIGger:{A|B}:PULse:TRANsition:POLarity:
CH<x>, 2-933
TRIGger:{A|B}:PULse:TRANsition:QUALify, 2-934
TRIGger:{A|B}:PULse:TRANsition:
THReshold?, 2-935
TRIGger:{A|B}:PULse:TRANsition:THReshold:
BOTh, 2-935
TRIGger:{A|B}:PULse:TRANsition:THReshold:
HIGH, 2-936
TRIGger:{A|B}:PULse:TRANsition:THReshold:
LOW, 2-936
TRIGger:{A|B}:PULse:TRANsition:WHEn, 2-937
TRIGger:{A|B}:PULse:WIDth?, 2-938
TRIGger:{A|B}:PULse:WIDth:HIGHLimit, 2-938
TRIGger:{A|B}:PULse:WIDth:LOWLimit, 2-939
TRIGger:{A|B}:PULse:WIDth:
LOWPASSfilter, 2-940
TRIGger:{A|B}:PULse:WIDth:POLarity, 2-940
TRIGger:{A|B}:PULse:WIDth:POLarity:
CH<x>, 2-941
TRIGger:{A|B}:PULse:WIDth:QUALify, 2-942
TRIGger:{A|B}:PULse:WIDth:WHEn, 2-943
TRIGger:{A|B}:PULse:WINdow?, 2-944
TRIGger:{A|B}:PULse:WINdow:EVENT, 2-944
TRIGger:{A|B}:PULse:WINdow:POLarity, 2-945
TRIGger:{A|B}:PULse:WINdow:POLarity:
CH<x>, 2-946
TRIGger:{A|B}:PULse:WINdow:QUALify, 2-946
TRIGger:{A|B}:PULse:WINdow:THReshold?, 2-947
TRIGger:{A|B}:PULse:WINdow:THReshold:
BOTh, 2-948
TRIGger:{A|B}:PULse:WINdow:THReshold:
HIGH, 2-948
TRIGger:{A|B}:PULse:WINdow:THReshold:
LOW, 2-949
TRIGger:{A|B}:PULse:WINdow:WIDTH, 2-949
TRIGger:{A|B}:READY?, 2-950
TRIGger:{A|B}:TYPe, 2-951
TRIGger:{A|B}:UPPerthreshold:CH<x>, 2-952
TRIGger:A, 2-953
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:
DIRection, 2-958
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:
FORMat, 2-958
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:
MODe, 2-959
TRIGger:A:BUS:CAN{:IDentifier|:ADDRess}:
VALue, 2-959
TRIGger:A:BUS:CAN:CONDition, 2-954
TRIGger:A:BUS:CAN:DATa:DIRection, 2-955
TRIGger:A:BUS:CAN:DATa:FORMat, 2-955
TRIGger:A:BUS:CAN:DATa:QUALifier, 2-956
TRIGger:A:BUS:CAN:DATa:SIZe, 2-956
TRIGger:A:BUS:CAN:DATa:VALue, 2-957
TRIGger:A:BUS:CAN:FRAMEtype, 2-957
TRIGger:A:BUS:DATa:FORMat, 2-960
TRIGger:A:BUS:DATa:VALue, 2-960
TRIGger:A:BUS:ETHERnet:CONDition, 2-961
TRIGger:A:BUS:ETHERnet:DATa:FORMat, 2-962
TRIGger:A:BUS:ETHERnet:DATa:OFFSet, 2-962
TRIGger:A:BUS:ETHERnet:DATa:SIZe, 2-963
TRIGger:A:BUS:ETHERnet:DATa:VALue, 2-964
TRIGger:A:BUS:ETHERnet:IPHeader:
DESTinationaddr:FORMat, 2-964
TRIGger:A:BUS:ETHERnet:IPHeader:
DESTinationaddr:VALue, 2-965
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:
FORMat, 2-966
TRIGger:A:BUS:ETHERnet:IPHeader:PROTOcol:
VALue, 2-967

TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:FORMat, 2-967
TRIGger:A:BUS:ETHERnet:IPHeader:SOUrceaddr:VALue, 2-968
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:FORMat, 2-972
TRIGger:A:BUS:ETHERnet:MAC{:LENgth|:TYPe}:VALue, 2-972
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:FORMat, 2-969
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:DESTination:VALue, 2-969
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:FORMat, 2-970
TRIGger:A:BUS:ETHERnet:MAC:ADDRess:SOUrce:VALue, 2-971
TRIGger:A:BUS:ETHERnet:QTAG:FORMat, 2-973
TRIGger:A:BUS:ETHERnet:QTAG:VALue, 2-974
TRIGger:A:BUS:ETHERnet:QUALifier, 2-974
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:FORMat, 2-975
TRIGger:A:BUS:ETHERnet:TCPHeader:ACKnum:VALue, 2-976
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:FORMat, 2-977
TRIGger:A:BUS:ETHERnet:TCPHeader:DESTinationport:VALue, 2-977
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:FORMat, 2-978
TRIGger:A:BUS:ETHERnet:TCPHeader:SEQnum:VALue, 2-979
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:FORMat, 2-980
TRIGger:A:BUS:ETHERnet:TCPHeader:SOUrceport:VALue, 2-980
TRIGger:A:BUS:FLEXRAY:CONDition, 2-981
TRIGger:A:BUS:FLEXRAY:CYCLEcount:FORMat, 2-982
TRIGger:A:BUS:FLEXRAY:CYCLEcount:QUALifier, 2-982
TRIGger:A:BUS:FLEXRAY:CYCLEcount:VALue, 2-983
TRIGger:A:BUS:FLEXRAY:DATa:FORMat, 2-983
TRIGger:A:BUS:FLEXRAY:DATa:OFFSet, 2-984
TRIGger:A:BUS:FLEXRAY:DATa:QUALifier, 2-984
TRIGger:A:BUS:FLEXRAY:DATa:SIZe, 2-985
TRIGger:A:BUS:FLEXRAY:DATa:VALue, 2-985
TRIGger:A:BUS:FLEXRAY:EOFTYPE, 2-986
TRIGger:A:BUS:FLEXRAY:ERRTYPE, 2-986
TRIGger:A:BUS:FLEXRAY:FRAMEType, 2-987
TRIGger:A:BUS:FLEXRAY:HEADER:CRC, 2-987
TRIGger:A:BUS:FLEXRAY:HEADER:CYCLECount, 2-988
TRIGger:A:BUS:FLEXRAY:HEADER:FRAMEID, 2-988
TRIGger:A:BUS:FLEXRAY:HEADER:INDBits, 2-989
TRIGger:A:BUS:FLEXRAY:HEADER:PAYLENgth, 2-989
TRIGger:A:BUS:FLEXRAY:IDentifier:FORMat, 2-990
TRIGger:A:BUS:FLEXRAY:IDentifier:QUALifier, 2-990
TRIGger:A:BUS:FLEXRAY:IDentifier:VALue, 2-991
TRIGger:A:BUS:I2C:ADDress:FORMat, 2-991
TRIGger:A:BUS:I2C:ADDress:MODe, 2-992
TRIGger:A:BUS:I2C:ADDress:TYPe, 2-993
TRIGger:A:BUS:I2C:ADDress:VALue, 2-993
TRIGger:A:BUS:I2C:CONDition, 2-994
TRIGger:A:BUS:I2C:DATa:DIRection, 2-995
TRIGger:A:BUS:I2C:DATa:FORMat, 2-995
TRIGger:A:BUS:I2C:DATa:SIZe, 2-996
TRIGger:A:BUS:I2C:DATa:VALue, 2-996
TRIGger:A:BUS:LIN:CONDition, 2-997
TRIGger:A:BUS:LIN:DATa:FORMat, 2-998
TRIGger:A:BUS:LIN:DATa:QUALifier, 2-998
TRIGger:A:BUS:LIN:DATa:SIZe, 2-999
TRIGger:A:BUS:LIN:DATa:VALue, 2-999
TRIGger:A:BUS:LIN:ERRTYPE, 2-1000
TRIGger:A:BUS:LIN:IDentifier:FORMat, 2-1000
TRIGger:A:BUS:LIN:IDentifier:VALue, 2-1001
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:FORMat, 2-1001
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:QUALifier, 2-1002
TRIGger:A:BUS:MIL1553B:COMMAND:ADDRess:VALue, 2-1003
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt, 2-1003
TRIGger:A:BUS:MIL1553B:COMMAND:COUNt:FORMat, 2-1004
TRIGger:A:BUS:MIL1553B:COMMAND:PARity, 2-1005

TRIGger:A:BUS:MIL1553B:COMMAND:
SUBADdress, 2-1005
TRIGger:A:BUS:MIL1553B:COMMAND:
SUBADdress:FORMat, 2-1006
TRIGger:A:BUS:MIL1553B:COMMAND:
TRBit, 2-1007
TRIGger:A:BUS:MIL1553B:CONDition, 2-1008
TRIGger:A:BUS:MIL1553B:DATA:FORMat, 2-1009
TRIGger:A:BUS:MIL1553B:DATA:PARity, 2-1010
TRIGger:A:BUS:MIL1553B:DATA:VALue, 2-1011
TRIGger:A:BUS:MIL1553B:ERRTYPE, 2-1011
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:
FORMat, 2-1012
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:
QUALifier, 2-1013
TRIGger:A:BUS:MIL1553B:STATUS:ADDRess:
VALue, 2-1013
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
BCR, 2-1014
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
BUSY, 2-1015
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
DBCA, 2-1016
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
INSTR, 2-1016
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
ME, 2-1017
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
PARity, 2-1018
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
SRQ, 2-1018
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
SUBSF, 2-1019
TRIGger:A:BUS:MIL1553B:STATUS:BIT:
TF, 2-1020
TRIGger:A:BUS:MIL1553B:TIME:
LESSLimit, 2-1021
TRIGger:A:BUS:MIL1553B:TIME:
MORELimit, 2-1021
TRIGger:A:BUS:MIL1553B:TIME:
QUALifier, 2-1022
TRIGger:A:BUS:PCIE:CHARacter:CHAR, 2-1023
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:
MINus, 2-1023
TRIGger:A:BUS:PCIE:CHARacter:SYMbol:
PLUS, 2-1024
TRIGger:A:BUS:PCIE:CONDition, 2-1025
TRIGger:A:BUS:PCIE:DISParity, 2-1025
TRIGger:A:BUS:PCIE:ERROR, 2-1026
TRIGger:A:BUS:PCIE:FORMat, 2-1026
TRIGger:A:BUS:PCIE:PATtern:CHAR<x>, 2-1027
TRIGger:A:BUS:PCIE:PATtern:ORDERedset, 2-1028
TRIGger:A:BUS:PCIE:PATtern:SYMbol:
MINus<x>, 2-1028
TRIGger:A:BUS:PCIE:PATtern:SYMbol:
PLUS<x>, 2-1029
TRIGger:A:BUS:RS232C:CONDition, 2-1030
TRIGger:A:BUS:RS232C:DATa:FORMat, 2-1030
TRIGger:A:BUS:RS232C:DATa:SIZe, 2-1031
TRIGger:A:BUS:RS232C:DATa:VALue, 2-1031
TRIGger:A:BUS:S8B10B:CHARacter:CHAR, 2-1032
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:
MINus, 2-1032
TRIGger:A:BUS:S8B10B:CHARacter:SYMbol:
PLUS, 2-1033
TRIGger:A:BUS:S8B10B:CONDition, 2-1034
TRIGger:A:BUS:S8B10B:DISParity, 2-1034
TRIGger:A:BUS:S8B10B:ERROR, 2-1035
TRIGger:A:BUS:S8B10B:FORMat, 2-1036
TRIGger:A:BUS:S8B10B:PATtern:
CHAR<x>, 2-1036
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:
MINus<x>, 2-1037
TRIGger:A:BUS:S8B10B:PATtern:SYMbol:
PLUS<x>, 2-1037
TRIGger:A:BUS:SOUrce, 2-1038
TRIGger:A:BUS:SPI:CONDition, 2-1038
TRIGger:A:BUS:SPI:DATa:FORMat, 2-1039
TRIGger:A:BUS:SPI:DATa:SIZe, 2-1040
TRIGger:A:BUS:SPI:DATa:VALue, 2-1040
TRIGger:A:BUS:USB:ADDress:FORMat, 2-1041
TRIGger:A:BUS:USB:ADDress:HIVALue, 2-1042
TRIGger:A:BUS:USB:ADDress:VALue, 2-1043
TRIGger:A:BUS:USB:CHARacter:CHAR, 2-1043
TRIGger:A:BUS:USB:CHARacter:SYMbol:
MINus, 2-1044
TRIGger:A:BUS:USB:CHARacter:SYMbol:
PLUS, 2-1044
TRIGger:A:BUS:USB:CONDition, 2-1045
TRIGger:A:BUS:USB:DATa:FORMat, 2-1046
TRIGger:A:BUS:USB:DATa:HIVALue, 2-1046
TRIGger:A:BUS:USB:DATa:OFFSet, 2-1047
TRIGger:A:BUS:USB:DATa:SIZe, 2-1047
TRIGger:A:BUS:USB:DATa:TYPe, 2-1048

TRIGger:A:BUS:USB:DATa:VALue, 2-1049
TRIGger:A:BUS:USB:DISParity, 2-1049
TRIGger:A:BUS:USB:ENDPoint:FORMat, 2-1050
TRIGger:A:BUS:USB:ENDPoint:HIVALue, 2-1051
TRIGger:A:BUS:USB:ENDPoint:VALue, 2-1051
TRIGger:A:BUS:USB:ERROR, 2-1052
TRIGger:A:BUS:USB:ERRTYPE, 2-1052
TRIGger:A:BUS:USB:FORMat, 2-1053
TRIGger:A:BUS:USB:HANDShaketype, 2-1054
TRIGger:A:BUS:USB:PATtern:CHAR<x>, 2-1054
TRIGger:A:BUS:USB:PATtern:NUMSymbols, 2-1055
TRIGger:A:BUS:USB:PATtern:ORDERedset, 2-1055
TRIGger:A:BUS:USB:PATtern:SYMbol:MINus<x>, 2-1056
TRIGger:A:BUS:USB:PATtern:SYMbol:PLUS<x>, 2-1056
TRIGger:A:BUS:USB:QUALifier, 2-1057
TRIGger:A:BUS:USB:SOF:FORMat, 2-1058
TRIGger:A:BUS:USB:SOF:FRAMENUMber, 2-1058
TRIGger:A:BUS:USB:SPECIALType, 2-1059
TRIGger:A:BUS:USB:SPLIT:ET:VALue, 2-1060
TRIGger:A:BUS:USB:SPLIT:HUB:FORMat, 2-1060
TRIGger:A:BUS:USB:SPLIT:HUB:VALue, 2-1061
TRIGger:A:BUS:USB:SPLIT:PORT:FORMat, 2-1062
TRIGger:A:BUS:USB:SPLIT:PORT:VALue, 2-1062
TRIGger:A:BUS:USB:SPLIT:SC:VALue, 2-1063
TRIGger:A:BUS:USB:SPLIT:SE:VALue, 2-1063
TRIGger:A:BUS:USB:TOKENType, 2-1064
TRIGger:A:CAN:CONDition, 2-1065
TRIGger:A:CAN:DATa:DIRection, 2-1066
TRIGger:A:CAN:DATa:LEVel, 2-1066
TRIGger:A:CAN:DATa:SOURce, 2-1067
TRIGger:A:CAN:DATa:VALue, 2-1067
TRIGger:A:CAN:FORMat, 2-1068
TRIGger:A:CAN:FRAMEtype, 2-1068
TRIGger:A:CAN:IDENTifier:MODe, 2-1069
TRIGger:A:CAN:IDENTifier:VALue, 2-1070
TRIGger:A:CAN:PROBE, 2-1070
TRIGger:A:CAN:SPEed, 2-1071
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:PULSEForm, 2-1071
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:HIGH, 2-1072
TRIGger:A:COMMunication:{AMI|HDB3|B3ZS|B6ZS|B8ZS}:THReshold:LOW, 2-1072
TRIGger:A:COMMunication:BITRate, 2-1073
TRIGger:A:COMMunication:CLOCk:POLarity, 2-1074
TRIGger:A:COMMunication:CMI:PULSEForm, 2-1074
TRIGger:A:COMMunication:CODe, 2-1075
TRIGger:A:COMMunication:SOURce, 2-1076
TRIGger:A:COMMunication:SOURce:TYPe, 2-1076
TRIGger:A:COMMunication:STANdard, 2-1077
TRIGger:A:EDGE:ENVelope, 2-1080
TRIGger:A:HOLDoff?, 2-1080
TRIGger:A:HOLDoff:ACTUal?, 2-1081
TRIGger:A:HOLDoff:BY, 2-1081
TRIGger:A:HOLDoff:TIMe, 2-1082
TRIGger:A:I2C:ADDRess:MODe, 2-1082
TRIGger:A:I2C:ADDRess:RWINClude, 2-1083
TRIGger:A:I2C:ADDRess:TYPe, 2-1083
TRIGger:A:I2C:ADDRess:VALue, 2-1084
TRIGger:A:I2C:CLOCk:LEVel, 2-1085
TRIGger:A:I2C:CLOCk:SOURce, 2-1085
TRIGger:A:I2C:CONDition, 2-1086
TRIGger:A:I2C:DATa:DIRection, 2-1086
TRIGger:A:I2C:DATa:LEVel, 2-1087
TRIGger:A:I2C:DATa:SOURce, 2-1087
TRIGger:A:I2C:DATa:STARt, 2-1088
TRIGger:A:I2C:DATa:VALue, 2-1088
TRIGger:A:I2C:FORMat, 2-1089
TRIGger:A:MODe, 2-1089
TRIGger:A:PULse:GLItch:ENVelope, 2-1090
TRIGger:A:PULse:PERiod?, 2-1090
TRIGger:A:PULse:PERiod:HIGHLimit, 2-1091
TRIGger:A:PULse:PERiod:LOWLimit, 2-1092
TRIGger:A:PULse:PERiod:POLarity, 2-1093
TRIGger:A:PULse:PERiod:QUALify, 2-1093
TRIGger:A:PULse:PERiod:VIEW, 2-1094
TRIGger:A:PULse:PERiod:WHEn, 2-1095
TRIGger:A:PULse:TIMEOut:ENVelope, 2-1096
TRIGger:A:PULse:WIDth:ENVelope, 2-1097
TRIGger:A:RS232:BAUd, 2-1097
TRIGger:A:RS232:DATa:LEVel, 2-1098
TRIGger:A:RS232:DATa:SOURce, 2-1098
TRIGger:A:RS232:DATa:VALue, 2-1099
TRIGger:A:RS232:FORMat, 2-1099
TRIGger:A:RS232:PARity, 2-1100

TRIGger:A:SERIAL:BITRate, 2-1100
TRIGger:A:SERIAL:CLOCk:LEVel, 2-1101
TRIGger:A:SERIAL:CLOCk:POLarity, 2-1102
TRIGger:A:SERIAL:CLOCk:SOUrce, 2-1102
TRIGger:A:SERIAL:CODe, 2-1103
TRIGger:A:SERIAL:DATa:FORMat, 2-1103
TRIGger:A:SERIAL:DATa:PATtern, 2-1104
TRIGger:A:SERIAL:DATa:PATtern:NRZ, 2-1105
TRIGger:A:SERIAL:DATa:PATtern:S8B10B, 2-1106
TRIGger:A:SERIAL:LOCKLen, 2-1107
TRIGger:A:SERIAL:LOCKOffset, 2-1107
TRIGger:A:SERIAL:SOUrce, 2-1108
TRIGger:A:SERIAL:STANdard, 2-1108
TRIGger:A:SERIAL:TRIGgeron, 2-1110
TRIGger:A:SPI:CONDition, 2-1110
TRIGger:A:SPI:DATa:MISO:ACTIVE, 2-1111
TRIGger:A:SPI:DATa:MISO:LEVel, 2-1111
TRIGger:A:SPI:DATa:MISO:SOUrce, 2-1112
TRIGger:A:SPI:DATa:MISO:VALue, 2-1112
TRIGger:A:SPI:DATa:MOSI:ACTIVE, 2-1113
TRIGger:A:SPI:DATa:MOSI:LEVel, 2-1113
TRIGger:A:SPI:DATa:MOSI:SOUrce, 2-1114
TRIGger:A:SPI:DATa:MOSI:VALue, 2-1114
TRIGger:A:SPI:DATa:STARt, 2-1115
TRIGger:A:SPI:FORMat, 2-1115
TRIGger:A:SPI:SCLK:ACTIVE, 2-1116
TRIGger:A:SPI:SCLK:LEVel, 2-1116
TRIGger:A:SPI:SCLK:SOUrce, 2-1117
TRIGger:A:SPI:SS:ACTIVE, 2-1117
TRIGger:A:SPI:SS:LEVel, 2-1118
TRIGger:A:SPI:SS:SOUrce, 2-1118
TRIGger:A:VIDeo?, 2-1119
TRIGger:A:VIDeo:CUSTom?, 2-1119
TRIGger:A:VIDeo:CUSTom:FORMat, 2-1120
TRIGger:A:VIDeo:CUSTom:LINEPeriod, 2-1120
TRIGger:A:VIDeo:CUSTom:SYNCInterval, 2-1121
TRIGger:A:VIDeo:FIELD, 2-1121
TRIGger:A:VIDeo:HOLdoff:FIELD, 2-1122
TRIGger:A:VIDeo:LINE, 2-1123
TRIGger:A:VIDeo:POLarity, 2-1124
TRIGger:A:VIDeo:SCAN, 2-1124
TRIGger:A:VIDeo:SOUrce, 2-1125
TRIGger:A:VIDeo:STANdard, 2-1126
TRIGger:AUXLevel, 2-1127
TRIGger:B, 2-1128
TRIGger:B:BY, 2-1128
TRIGger:B:EDGE:ENVelope, 2-1129
TRIGger:B:EVENTS?, 2-1129
TRIGger:B:EVENTS:COUNt, 2-1130
TRIGger:B:PULse:GLItch:ENVelope, 2-1130
TRIGger:B:PULse:PERiod?, 2-1131
TRIGger:B:PULse:PERiod:HIGHLimit, 2-1132
TRIGger:B:PULse:PERiod:LOWLimit, 2-1133
TRIGger:B:PULse:PERiod:POLarity, 2-1133
TRIGger:B:PULse:PERiod:QUAlify, 2-1134
TRIGger:B:PULse:PERiod:VIEW, 2-1135
TRIGger:B:PULse:PERiod:WHEn, 2-1136
TRIGger:B:PULse:TIMEOut:ENVelope, 2-1137
TRIGger:B:PULse:WIDth:ENVelope, 2-1137
TRIGger:B:RESET:ACCEPTCOUNT?, 2-1138
TRIGger:B:RESET:ACCEPTTIMEout, 2-1138
TRIGger:B:RESET:RATIO?, 2-1139
TRIGger:B:RESET:RATIOENable, 2-1140
TRIGger:B:RESET:REJECTCOUNT?, 2-1141
TRIGger:B:RESET:REJECTTIMEout, 2-1141
TRIGger:B:RESET:SOUrce, 2-1142
TRIGger:B:RESET:STATE, 2-1143
TRIGger:B:RESET:THReshold, 2-1143
TRIGger:B:RESET:TIMEOut, 2-1144
TRIGger:B:RESET:TOTALCOUNT?, 2-1144
TRIGger:B:RESET:TRANsition, 2-1145
TRIGger:B:RESET:TYPe, 2-1146
TRIGger:B:SCAN:ADVANCEafter, 2-1147
TRIGger:B:SCAN:ENAble, 2-1148
TRIGger:B:SCAN:ENDevent, 2-1148
TRIGger:B:SCAN:MODE, 2-1149
TRIGger:B:SCAN:STARTevent, 2-1149
TRIGger:B:STATE, 2-1150
TRIGger:B:TIMe, 2-1151
TRIGger:ENHanced, 2-1151
TRIGger:EQUation<1..1>, 2-1152
TRIGger:LVLSrcpreference, 2-1153
TRIGger:QUALification:BUS:FORMat, 2-1153
TRIGger:QUALification:BUS:SOUrce, 2-1154
TRIGger:QUALification:BUS:VALue, 2-1154
TRIGger:SENSITivity, 2-1155
TRIGger:SHOWEQuation, 2-1156
TRIGger:STATE?, 2-1156
*TST?, 2-1157

U

UNLock, 2-1157
USBTMC:PRODUCTID:DECimal?, 2-1158
USBTMC:PRODUCTID:HEXadecimal?, 2-1158

USBTMC:SERIALnumber?, 2-1158
USBTMC:VENDORID:DECimal?, 2-1159
USBTMC:VENDORID:HEXadecimal?, 2-1159

V

VERBose, 2-1159
VISual?, 2-1160
VISual:AREA<x>?, 2-1161
VISual:AREA<x>:DISplay, 2-1161
VISual:AREA<x>:FLIP, 2-1162
VISual:AREA<x>:HEIGHT, 2-1162
VISual:AREA<x>:LABel, 2-1163
VISual:AREA<x>:OPERATION<1..1>, 2-1163
VISual:AREA<x>:ROTAtion, 2-1164
VISual:AREA<x>:SHAPE, 2-1164
VISual:AREA<x>:SHOWLOGic, 2-1165
VISual:AREA<x>:SOURCE<1..1>, 2-1165
VISual:AREA<x>:VERTICES, 2-1166
VISual:AREA<x>:WIDTH, 2-1166
VISual:AREA<x>:XPOSition, 2-1167
VISual:AREA<x>:YPOSition, 2-1167
VISual:AREACOLOr, 2-1160
VISual:ASPECTratio, 2-1168
VISual:DELETEAREA, 2-1168
VISual:ENAble, 2-1169
VISual:FILE:RECALL, 2-1169
VISual:FILE:SAVE, 2-1170

W

*WAI, 2-1170
WAVFrm?, 2-1171
WAVFRMStream?, 2-1172
WFMInpre?, 2-1173
WFMInpre:BIT_Nr, 2-1173
WFMInpre:BN_Fmt, 2-1174
WFMInpre:BYT_Nr, 2-1174
WFMInpre:BYT_Or, 2-1175
WFMInpre:ENCdg, 2-1175
WFMInpre:NR_FR?, 2-1176
WFMInpre:NR_Pt, 2-1176
WFMInpre:PT_Fmt, 2-1177
WFMInpre:PT_Off, 2-1178
WFMInpre:WFId, 2-1178
WFMInpre:XINcr, 2-1179
WFMInpre:XUNit, 2-1179
WFMInpre:XZEro, 2-1180
WFMInpre:YMUlt, 2-1180
WFMInpre:YOFf, 2-1181
WFMInpre:YUNit, 2-1182
WFMInpre:YZEro, 2-1182
WFMOutpre?, 2-1183
WFMOutpre:BIT_Nr, 2-1183
WFMOutpre:BN_Fmt, 2-1184
WFMOutpre:BYT_Nr, 2-1184
WFMOutpre:BYT_Or, 2-1185
WFMOutpre:ENCdg, 2-1185
WFMOutpre:NR_FR?, 2-1186
WFMOutpre:NR_Pt?, 2-1186
WFMOutpre:PT_Fmt?, 2-1187
WFMOutpre:PT_Off?, 2-1187
WFMOutpre:PT_ORder?, 2-1188
WFMOutpre:WFId?, 2-1188
WFMOutpre:XINcr?, 2-1190
WFMOutpre:XUNit?, 2-1190
WFMOutpre:XZEro?, 2-1190
WFMOutpre:YMUlt?, 2-1191
WFMOutpre:YOFf?, 2-1191
WFMOutpre:YUNit?, 2-1192
WFMOutpre:YZEro?, 2-1192

Z

ZOOm, 2-1193
ZOOm:GRAticule:SIZE, 2-1193
ZOOm:HORizontal:POSition, 2-1194
ZOOm:HORizontal:SCAle, 2-1194
ZOOm:MATH<x>:HORizontal:POSition, 2-1195
ZOOm:MATH<x>:HORizontal:SCAle, 2-1195
ZOOm:MATH<x>:VERTical:POSition, 2-1196
ZOOm:MATH<x>:VERTical:SCAle, 2-1196
ZOOm:MODe, 2-1197
ZOOm:REF<x>:HORizontal:POSition, 2-1197
ZOOm:REF<x>:HORizontal:SCAle, 2-1198
ZOOm:REF<x>:VERTical:POSition, 2-1198
ZOOm:REF<x>:VERTical:SCAle, 2-1199
ZOOm:SCROLL:DIREction, 2-1199
ZOOm:SCROLL:LOCk, 2-1200
ZOOm:SCROLL:SPEED, 2-1200
ZOOm:STATE, 2-1201
ZOOm:VERTical:POSition, 2-1201
ZOOm:VERTical:SCAle, 2-1202
ZOOm:ZOOM<x>, 2-1202
ZOOm:ZOOM<x>:<wfm>:DISplay, 2-1204

ZOOm:ZOOM<x>:<wfm>:HORizontal:POSition, 2-1205
ZOOm:ZOOM<x>:<wfm>:HORizontal:SCAle, 2-1206
ZOOm:ZOOM<x>:<wfm>:VERTical:POSition, 2-1206
ZOOm:ZOOM<x>:<wfm>:VERTical:SCAle, 2-1207
ZOOm:ZOOM<x>:DCHAN:DISplay, 2-1203
ZOOm:ZOOM<x>:SCROLLLock, 2-1203
ZOOm:ZOOM<x>:STATE, 2-1204