Tektronix®

8 Series Sampling Oscilloscope Programmer Manual

This document supports TSOVu software release v1.4

www.tek.com

077-1609-03

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.

TekVISA is a trademark of Tektronix, Inc.

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

This programmer manual provides you with the information required to use SCPI programmatic commands (PI) to remotely control a Tektronix TSO820 Sampling Oscilloscope through LAN connection.

Getting Started

This programmer manual provides you with the information required to use PI commands to remotely control your instrument. With this information, you can write computer programs that will perform virtually all the same functions as provided by the User Interface.

The programmer manual is divided into the following major sections:

- **Syntax and Commands**. This section provides an overview of the command syntax used 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.
- **Commands**. This section contains all the commands and related arguments, returns, and examples. Commands are listed by group.
- **Status and Events**. This section 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 discussed include registers, queues, event handling sequences, synchronization methods, and messages that the instrument may return, including error messages.

Command syntax

You can control the operations and functions of the instrument through the LAN interface using commands and queries. The related topics listed below describe the syntax of these commands and queries. The topics also describe the conventions that the 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 documentation describes the commands and queries using Backus-Naur Form (BNF) notation. Refer to the following table for the symbols that are used.

Table 2-1: Symbols for Backus-Naur Form

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

Command and Query structure

Commands consist of set commands and query commands (usually called commands and queries). Commands modify 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 needs to execute the command or query. Command messages may contain five element types, defined in the following table.

Table 2-2: Command Message Elements

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 a star (*).
<Mnemonic>	This is a header subfunction. Some command headers have only one mnemonic. If a command header has multiple mnemonics, a colon (:) character always separates them from each other.
<Argument>	This is a quantity, quality, restriction, or limit associated with the header. Some commands have no arguments while others have multiple arguments. A <space> separates arguments from the header. A <comma> separates arguments from each other.
<Comma>	A single comma is used between arguments of multiple-argument commands. Optionally, there may be white space characters before and after the comma.
<Space>	A white space character is used between a command header and the related argument. Optionally, a white space may consist of multiple white space characters.

Commands

Commands cause the 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 may 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> [<Coma><Argument>]...]`

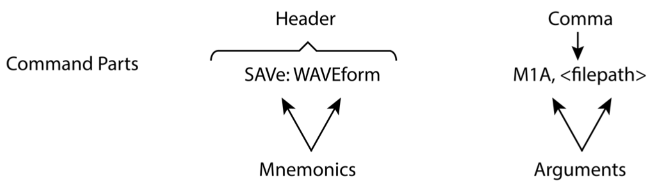
You can specify a query command at any level within the command tree unless otherwise noted. These branch queries return information about all the mnemonics below the specified branch or level. For example, HIStogram:STATistics:STDdev? returns the standard deviation of the histogram, while HIStogram:STATistics? returns all the histogram statistics, and HIStogram? returns all the histogram parameters.

Headers

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

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

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



Clearing the Instrument

You can clear the Output Queue and reset TSOVu to accept a new command or query by using the selected Device Clear (DCL) GPIB function. Refer to your GPIB library documentation for further details about the selected 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. Each command in this documentation shows the abbreviations in capitals. For example, you can enter the command ACQuire:NUMAvg simply as ACQ:NUMAVG or acq:numavg.

Abbreviation rules may 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 commands and queries, you must follow these rules:

- Separate completely different headers by a semicolon and by the beginning colon on all commands except the first one. For example, the commands `TRIGger:SOUrce FREErun` and `ACQuire:NUMAVg 10`, can be concatenated into the following single command:

  ```
  TRIGger:SOUrce FREErun;:ACQuire:NUMAVg 10
  ```

- If concatenated commands have headers that differ by only the last mnemonic, you can abbreviate the second command and eliminate the beginning colon. For example, you can concatenate the commands `ACQuire:MODe AVERage` and `ACQuire:NUMAVg 10` into a single command:

  ```
  ACQuire:MODe AVERage; NUMAVg 10
  ```

  The longer version works equally well:

  ```
  ACQuire:MODe AVERage;:ACQuire:NUMAVg 10
  ```

- Never precede a star (*) command with a colon:

  ```
  ACQuire:MODe AVERage;*OPC
  ```

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

- When you concatenate queries, the responses to all the queries are concatenated into a single response message. For example, if the Acquire mode is set to sample and state is set to on, the concatenated query `:ACQuire:MODe?;STATE?` will return the following.

  If the header is on:

  ```
  :ACQuire:MODe SAMple :ACQuire:STATE ON
  ```

  If the header is off:

  ```
  SAMple;ON
  ```

- Set commands and queries may be concatenated in the same message. For example,

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

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

Here are some invalid concatenations:

`DISplay:MODE TILE;ACQuire:NUMAVg 10` (no colon before ACQuire)

`DISPLAY:REF1 1 1;::REF2 0` (extra colon before `REF2;` use `DISPLAY:REF1 1;REF2 0` instead)

`DISplay:MODE TILE;:*OPC` (colon before a star (*) command)

`CURSOR:VIEW1:VBARS:POSITION1 21E-9;VBARS:POSITION2 3.45E-6` (levels of the mnemonics are different; either remove the second use of `VBARS` or place `:CURSOR:VIEW1:` in front of `VBARS:POSITION2 3.45E-6`)

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. It allows white space before the terminator. For example, `CR LF`.

Constructed mnemonics

Some header mnemonics specify one of a range of mnemonics. A channel mnemonic has to be M<n>{A|B}, where <n> is the module number and {A|B} is the channel name of the module. You use these mnemonics in the command just as you do any other mnemonic. For example, there is a M1A:POSITION command, and there is also a M1B:POSITION command.

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<n>	A cursor selector; <n> is either 1 or 2.
POSITION<n>	A cursor selector; <n> is either 1 or 2.
HPOS<n>	A cursor selector; <n> is either 1 or 2.

Measurement specifier mnemonics

Commands can specify which measurement to set or query as a mnemonic in the header. Up to 32 automated measurements may be displayed in the system. The displayed measurements are specified in this way:

Table 2-6: Measurement specifier mnemonics

Symbol	Meaning
MEAS<n>	A measurement specifier; <n> is 1 through 32.
SOURCE<n>	A waveform specifier; <n> is either 1 (Source 1 waveform) or 2 (Source 2 waveform).
REFLevel<n>	A waveform specifier for reference level measurements; <n> is either 1 (Source 1 waveform) or 2 (Source 2 waveform).
GATE<n>	A gate specifier; <n> is either 1 (Gate 1) or 2 (Gate 2).

Channel mnemonics

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

Table 2-7: Channel mnemonics

Symbol	Meaning
M<n>{A|B}	A channel specifier; <n> is 1 to 4.

Reference waveform mnemonics

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

Table 2-8: Reference waveform mnemonics

Symbol	Meaning
REF<n>	A reference waveform specifier; <n> is 1 through 8.

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 2-9: 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 2-10: Quoted String Argument

Symbol	Meaning
<QString>	Quoted string of ASCII text

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

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

Here are some invalid strings:

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

Block

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

Table 2-11: Block Argument

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

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

Command Groups and Descriptions

Acquisition Command Group

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

Using these commands for acquiring waveforms, you can do the following:

- Start and stop acquisitions.
- Control whether all waveforms are simply acquired and averaged.
- Set the controls or conditions that start and stop acquisitions.
- Get data on acquired waveforms and histograms.
- Get acquisition parameters.
- Clear all acquired data.

ACQuire:STOPAfter:CONDition		
1. ACQWfms	ACQuire: STOPAfter: COUNT	
2. AVGComp	ACQuire:NUMAVg	
ACQuire:RAAFter (EACQuisition | STOP | COUNt)		Sets/queries the "run analysis after" state of the instrument.
1. EACQuisition		
2. STOP		
3. COUNt		
ACQuire:RAAFter:COUNt <NR3>		Sets/queries periodicity at which analysis occurs
ACQuire:STATE \\{ OFF | ON | RUN | STOP | <NR1> }*		RUN/STOP button on the right side of TSOVu
ACQuire:MODe { SAMple | AVERage }		Acquisition Mode in "Acquisition Menu"
1. SAMple		
2. AVERage	ACQuire:NUMAVg	
ACQuire:CURRentcount:ACQWfms	smaller than Stop After Count	
ACQuire:DATA:CLEar		Clear acquisition data, but not setup

ACQuire:MODe

Description
This command sets or queries the acquisition mode of the instrument, which determines how the final value of the acquisition interval is generated from the many data samples. The instrument applies the specified mode globally to all channel waveforms that it acquires.
The three, mutually exclusive acquisition modes are:

- Sample: Use Sample mode to see the signal in its purest form with no post processing. This is the default mode.
- Average: Use Average mode to reduce the apparent noise in the signal to reveal fundamental waveform behavior.

Syntax
ACQuire:MODe { SAMple | AVERage }
ACQuire:MODe?

Related Commands
ACQuire:NUMAVg

Arguments

- SAMple specifies sample mode, in which the displayed data point value is simply the sampled value that was taken during the acquisition interval. There is no post processing of acquired samples; the instrument overwrites waveforms at each new acquisition cycle. SAMple is the default acquisition mode.
- AVERage specifies averaging mode, in which the resulting waveform shows an average of SAMple data points from several consecutive waveform acquisitions. The instrument processes the number of waveforms you specify into the acquired waveform, creating a running back-weighted exponential average of the input signal. The number of waveform acquisitions that go into making up the average waveform is set or queried using the ACQuire:NUMAVg command.

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

Examples
ACQUIRE:MODE AVERage sets the acquisition mode to display a waveform that is an average of SAMple data points from several consecutive waveform acquisitions.

ACQuire:RAAFter

Description
This command sets or queries the run analysis after state of the instrument. Analysis includes:

- Measurements
- Histograms
- Mask tests

Syntax
ACQuire:RAAFter { EACQuisition | STOP | COUNt }
ACQuire:RAAFter?

Arguments

- EACQuistion shall set the analysis to be run on every acquisition
- STOP shall set the analysis to be run on acquisition stop

- COUNT shall set the analysis to be run periodically as determined by the count specified

Related Commands
ACQuire:RAAFter:COUNt

Returns
The current run analysis mode (EACQuisition, STOP, COUNT).

Examples
ACQUIRE:RAAFTER COUNT sets the analysis to be run after a specific number of acquisitions.
ACQUIRE:RAAFTER? may return EACQUISITION, indicating that analysis runs after every acquisition.

ACQuire:RAAFter:COUNt

Description
This command sets or queries the run analysis count parameter of the acquisition to indicate the periodicity at which analysis occurs.

Syntax
ACQuire:RAAFter:COUNt <NR3>
ACQuire:RAAFter:COUNt?

Arguments
<NR3> indicates the periodicity of the analysis with respect to acquisitions.

Related Commands
ACQuire:RAAFter

Returns
<NR3> indicating the periodicity of the analysis with respect to acquisitions.

Examples
ACQUIRE:RAAFTER:COUNT 23 sets the analysis to be run for 23 acquisitions.
ACQUIRE:RAAFTER? may return 11, indicating that analysis will run for 11 acquisitions.

Further information
If acquisition stop on condition is set to 53 and analysis count periodicity is set to 5, then 11 analyses will run, provided measurements have been added before acquisitions have started.

ACQuire:STATE

Description
This command starts or stops acquisitions or queries whether the acquisition is running or stopped.

Syntax
ACQuire:STATE { OFF | ON | RUN | STOP | 1 | 0 }
ACQuire:STATE?

Arguments
- OFF stops acquisitions.
- STOP stops acquisitions.
- ON starts acquisitions.
- RUN starts acquisitions.

- 0 stops acquisitions.
- 1 starts acquisitions.

Returns
ACQUIRE:STATE? might return ACQUIRE:STATE 1, indicating that the acquisition system is running.

Examples
ACQUIRE:STATE RUN starts acquisition of waveform data.

ACQuire:CURRentcount:ACQWfms?

Description
This query only command returns the current count value of acquired waveforms. The target value of this count is set by the ACQuire:STOPAfter:COUNt command (in conjunction with the ACQuire:STOPAfter:CONDition command). The instrument then counts up to this value. When the count reaches (or exceeds) the value, acquisition stops, and the specified StopAfter action is enabled.

Syntax
ACQuire:CURRentcount:ACQWfms?

Related Commands
- ACQuire:STOPAfter:COUNt
- ACQuire:STOPAfter:CONDition

Arguments
Query only command has no arguments.

Returns
NR1 is the current count value of acquired waveforms.

Examples
ACQUIRE:CURRENTCOUNT:ACQWFMS? might return ACQUIRE:CURRENTCOUNT:ACQWFMS 20, indicating that currently 20 waveforms have been acquired.

ACQuire:STOPAfter:MODe

Description
This command tells the instrument when to stop taking acquisitions. The query form of this command returns the StopAfter mode.
Syntax
ACQuire:STOPAfter:MODe { RUNSTop | CONDition }
ACQuire:STOPAfter:MODe?

Related Commands
ACQuire:STOPAfter:CONDition
ACQuire:STATE

Arguments
- RUNSTop specifies that the run and stop state is determined by the RUN/STOP button of the application.
- CONDition specifies that the run and stop state of the system is determined by a set a qualifiers specified by the StopAfter Condition. These sub-states are further described in the ACQuire:STOPAfter:CONDition section. (The instrument can still be stopped unconditionally by pressing the RUN/STOP button of the application or by sending the ACQuire:STATEcommand.)

Examples

- ACQUIRE:STOPAFTER:MODE RUNSTOP sets the instrument to run or stop acquisitions when the user presses the RUN/STOP button of the application or the user sends the ACQuire:STATE command.
- ACQUIRE:STOPAFTER:MODE? might return ACQUIRE:STOPAFTER:MODE CONDITION, indicating that the run and stop state of the system is determined by a set of qualifiers specified by the StopAfter condition.

ACQuire:STOPAfter:CONDition

Description
This command sets or queries the StopAfter condition. The StopAfter condition qualifies a stop condition for the acquisition system. Only one StopAfter condition can be active at a given time. Each StopAfter condition identifies, directly or indirectly, a specific data element or operation such that all mutually exclusive conditions are unique and unambiguous. This command allows you to specify the condition on which to stop acquiring. The condition is valid when the ACQuire:STOPAfter:MODE is set to CONDition.

Syntax
ACQuire:STOPAfter:CONDition { ACQWfms | AVGComp }
ACQuire:STOPAfter:CONDition?

Related Commands
ACQuire:STOPAfter:COUNt
ACQuire:NUMAVg

Arguments

- ACQWfms sets the instrument to stop acquiring after some specified number of raw acquisition cycles. This setting tells the instrument to count the number of MainTime base sweeps (Mag sweeps are not counted independently) and stop acquisition after the specified number of acquisitions has been reached. Use the ACQuire:STOPAfter:COUNt command to set the target number of waveforms.
- AVGComp sets the instrument to stop acquisition after the number of waveforms specified by the ACQuire:NUMAVg command have been acquired and averaged.

Examples
ACQUIRE:STOPAFTER:CONDITION ACQWFMS sets the instrument to stop acquiring after some specified number of raw acquisition cycles.
ACQUIRE:STOPAFTER:CONDITION? might return ACQUIRE:STOPAFTER:CONDITION ACQWFMS

ACQuire:STOPAfter:COUNt

Description
This command sets or queries the target StopAfter count for the condition specified by the ACQuire:STOPAfter:CONDition command. The current count for the condition must be equal to or greater than this value before acquisitions are stopped and a StopAfter action is enabled. The state of the numeric StopAfter count for each condition is kept individually so that you do not need to re-enter a count when switching between conditions. Use the appropriate ACQuire:CURRentcount command to get the current count for a condition.

Syntax
ACQuire:STOPAfter:COUNt <NR1>
ACQuire:STOPAfter:COUNt?

Related Commands
ACQuire:STOPAfter:MODe
ACQuire:STOPAfter:CONDition
ACQuire:CURRentcount:ACQWfms?

Arguments
NR1 is the count value that must be reached (or exceeded) before the acquisitions stop and StopAfter action can occur.

Examples
- ACQUIRE:STOPAFTER:COUNT 12 sets the StopAfter count for the specified condition to 12.
- ACQUIRE:STOPAFTER:COUNT? might return ACQuire:STOPAfter:COUNt 5, indicating that the total count for the specified condition is 5.

ACQuire:NUMAVg

Description
This command sets or queries the number of waveform acquisitions that makeup an averaged waveform. Use the ACQuire:MODe command to enable the Average mode.

Syntax
ACQuire:NUMAVg <NR1>
ACQuire:NUMAVg?

Related Commands
ACQuire:MODe

ACQuire:STOPAfter:CONDition

Arguments
NR1 is the number of consecutive waveform acquisitions (from 2 to 4,096) used for averaging.

Examples
- ACQUIRE:NUMAVG 10 specifies that an averaged waveform will show the result of combining 10 separately acquired waveforms.
- ACQUIRE:NUMAVG? might return ACQUIRE:NUMAVG 75, indicating that there are 75 acquisitions specified for averaging.

ACQuire:DATa:CLEar

Description
This command (no query form) causes an acquisition reset and clears all acquired data and clears the display. When a clear data occurs, it has the following effects:
- When Acquisition is Running the current waveform data is replaced by the waveform data of the next acquisition cycle when it is available.
- Counts. Resets all counts, including number of acquired waveforms, acquisition and average counts, conditional stop counts.
- Measurement statistics. Measurement statistics are reset.
- Histogram data and statistics. The data and all statistics will be cleared immediately.

Syntax
ACQuire:DATa:CLEar

Examples

ACQUIRE:DATA:CLEAR causes an acquisition reset and clears all acquired data.

Compensation Command Group

The compensation commands provide information about the current state of the compensation for the mainframe and all installed module channels, means to invoke compensation functions, and management of compensation storage memory locations.

COMPensate:M[n]{A|B}

Description
This command (no query form) compensates the module channel for DC variances.
Volatile run-time compensation data for compensated channels are saved into their respective nonvolatile user memories.

> **Warning: Before proceeding, please save your setup.**
> **For Mainframe SPC (Signal Path Compensation):**
> 1. **Disconnect or disable signals to mainframe's Clock Prescale Input.**
>
> **For Module SPC (Signal Path Compensation):**
> 1. **Leave any trigger/clock signal connected to the mainframe's Clock Prescale Input.**
> 2. **Disconnect or disable signals from sampling modules Inputs.**
> 3. **Terminate all unused electrical inputs with a 50 Ohms terminator and cover the unused optical modules inputs with dust covers.**

Syntax
COMPensate:M[n]{A|B}

Examples

COMPENSATE:M1A performs the compensation routines for channel A on module 1.

COMPensate:MAInframe

Description
This command (no query form) compensates the mainframe for DC variances.
Volatile run-time compensation data for compensated mainframes are saved into their respective nonvolatile user memories.

> **Warning: Before proceeding, please save your setup.**
> **For Mainframe SPC (Signal Path Compensation):**
> 2. **Disconnect or disable signals to mainframe's Clock Prescale Input.**
>
> **For Module SPC (Signal Path Compensation):**
> 4. **Leave any trigger/clock signal connected to the mainframe's Clock Prescale Input.**
> 5. **Disconnect or disable signals from sampling modules Inputs.**
> 6. **Terminate all unused electrical inputs with a 50 Ohms terminator and cover the unused optical modules inputs with dust covers.**

Syntax
COMPensate:MAInframe

Examples

COMPENSATE:MAINFRAME performs the compensation routines for channel A on module 1.

COMPensate:DATE:M[n]{A|B}?

Description
This is a query only command that returns the date and the time of the current in-use (that is, run-time) compensation data for the module channel.

Syntax
COMPensate:DATE:M[n]{A|B}?

Returns
<QString> Date and the time of the current in-use compensation data

Examples
COMPENSATE:DATE:M1A? might return COMPENSATE:DATE:M1A "10/15/2019 7:55:01 AM"

COMPensate:DATE:MAInframe?

Description
This is a query only command that returns the date and the time of the current in-use (that is, run-time) compensation data for the mainframe.

Syntax
COMPensate:DATE:MAInframe?

Returns
<QString> Date and the time of the current in-use (that is, run-time) compensation data for the mainframe.

Examples
COMPENSATE:DATE:MAINFRAME? might return COMPENSATE:DATE:MAINFRAME "12/23/1973 1:13:34 AM"

COMPensate:RESults?

Description
This is a query only command that returns an abbreviated status about the results of the last compensation execution. Any result other than PASS generally indicates a failure. For a more detailed message about the results of the last compensation execution, use the COMPensate:RESults:VERBose? query.

Syntax
COMPensate:RESults?

Returns
<QString>

Examples
COMPENSATE:RESULTS? might return COMPENSATE:RESULTS "PASS", indicating that the compensation was successful.

COMPensate:STATus:M[n]{A|B}?

Description

This is a query only command that returns the current compensation status for the module channel.

Syntax
COMPensate:STATus:M[n]{A|B}?

Returns
Enum. Possible responses are DEFaults, WARMup, FAIL, PASS, and COMPReq.

Examples
COMPENSATE:STATUS:M1A? might return COMPENSATE:STATUS:M1A COMPREQ, indicating that the warm-up period for the instrument has elapsed, but the current compensation temperature delta is greater than desired, or that the specified module has been moved to a different module compartment since last compensated. In either case, the instrument should be compensated again.

COMPensate:STATus:MAInframe?

Description
This is a query only command that returns the current compensation status for the mainframe.

Syntax
COMPensate:STATus:MAInframe?

Returns
Enum. Possible responses are DEFaults, WARMup, FAIL, PASS, and COMPReq.

Examples
COMPENSATE:STATUS:MAINFRAME? might return COMPENSATE:STATUS:MAINFRAME PASS, indicating that the current compensation data should allow the instrument to meet operating specifications.

COMPensate:TEMPerature:M[n]{A|B}?

Description
This query only command returns the difference (in °C) between the current temperature of the module channel and the associated temperature currently residing in the in-use run-time compensation memory.

Syntax
COMPensate:TEMPerature:M[n]{A|B}?

Returns
NR3

Examples
COMPENSATE:TEMPERATURE:M1A? might return COMPENSATE:TEMPERATURE:M1A 1.5

COMPensate:TEMPerature:MAInframe?

Description
This query only command returns the difference (in °C) between the current temperature of the mainframe and the associated temperature currently residing in the in-use run-time compensation memory.

Syntax

COMPensate:TEMPerature:MAInframe?

Returns
NR3

Examples
COMPENSATE:TEMPERATURE:MAINFRAME? might return
COMPENSATE:TEMPERATURE:MAINFRAME 2.7.

Calibration Command Group

The calibration commands provide information about the current state of the calibration for the mainframe and all resident sampling-module channels.

CALibration:TEMPerature:MAInframe?

Description

This query only command returns the difference (in °C) between the current temperature of the mainframe and the associated temperature currently residing in the in-use run-time calibration memory.

Syntax
CALibration:TEMPerature:MAInframe?

Returns
NR3

Examples
CALiBRATION:TEMPERATURE:MAINFRAME? might return
CALiBRATION:TEMPERATURE:MAINFRAME 2.7.

CALibration:TEMPerature:M[n]{A|B}?

Description

This query only command returns the difference (in °C) between the current temperature of the module channel and the associated temperature currently residing in the in-use run-time calibration memory.

Syntax
CALibration:TEMPerature:M[n]{A|B}?

Returns
NR3

Examples
CALiBRATION:TEMPERATURE:M1A? might return CALiBRATION:TEMPERATURE:M1A 1.5

CALibration:STATus:M[n]{A|B}?

Description

This is a query only command that returns the current calibration status for the module channel.

Syntax
CALibration:STATus:M[n]{A|B}?

Returns
Enum. Possible responses are FAIL or PASS.

Examples
CALiBRATION:STATUS:M1A? might return CALiBRATION:STATUS:M1A PASS indicates the calibration test has passed.

CALibration:STATus:MAInframe?

Description

This is a query only command that returns the current calibration status for the mainframe.

Syntax
CALibration:STATus:MAInframe?

Returns
Enum. Possible responses are FAIL and PASS

Examples
CALiBRATION:STATUS:MAINFRAME? might return CALiBRATION:STATUS:MAINFRAME PASS, indicating that the current calibration data should allow the instrument to meet operating specifications.

CALIbration:DATE:M[n]{A|B}?

Description
This is a query only command that returns the date and the time of the current in-use (that is, run-time) calibration data for the module channel.

Syntax
CALIbration:DATE:M[n]{A|B}?

Returns
<QString> Date and the time of the current in-use calibration data

Examples
CALIBRATION:DATE:M1A? might return :CALIBRATION:DATE:M1A "12/23/1973 1:13:34 AM"

CALIbration:DATE:MAInframe?

Description
This is a query only command that returns the date and the time of the current in-use (that is, run-time) calibration data for the mainframe.

Syntax
CALIbration:DATE:MAInframe?

Returns
<QString> Date and the time of the current in-use (that is, run-time) calibration data for the mainframe.

Examples
CALIBRATION:DATE:MAINFRAME? might return :CALIBRATION:DATE:MAINFRAME "12/23/1973 1:13:34 AM"

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**. Displays waveform cursors, which provide horizontal and vertical 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).

CURSor[:VIEW[x]]:CURSor[x]:SOUrce

Description
This command sets or queries which waveform is associated with the specified cursor. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate. The cursor is specified by x in the :CURSor[x] portion of the command, which can be 1 or 2.

Syntax
:CURSor[:VIEW[x]]:CURSor[x]:SOUrce { M[n]{A|B} | REF[x] }
:CURSor[:VIEW[x]]:CURSor[x]:SOUrce?

Arguments

- M[n]{A|B} specifies a live waveform to use as the source for the specified cursor.
- REF[x] specifies a reference waveform to use as the source for the specified cursor.

Returns
The waveform that is associated with the specified cursor.

Examples

- :CURSOR:CURSOR2:SOURCE M1B associates cursor 2 in the default waveform view with the module 1 channel B waveform.
- :CURSOR:VIEW2:CURSOR1:SOURCE? might return :CURSOR:VIEW2:CURSOR1:SOURCE

  REF5, indicating that in waveform view 2, cursor 1 is associated with the Ref 5 waveform.

CURSor[:VIEW[x]]:FUNCtion

Description
This command sets or queries cursor type. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate. Waveform cursors are not supported with Pattern Sync off.

Syntax
:CURSor[:VIEW[x]]:FUNCtion { WAVEform | VBArs | HBArs | VHBars }
:CURSor[:VIEW[x]]:FUNCtion?

Arguments

- WAVEform enables waveform cursors, which provide both vertical and horizontal unit readouts but are constrained to valid data points of the selected waveform.

- VBArs enables vertical bar cursors, which provide horizontal unit readouts.
- HBArs enables horizontal bar cursors, which provide vertical unit readouts.
- VHBars enables vertical and horizontal bar cursors, which provide their respective unit readouts.

Returns
The current cursor type

Examples
- :CURSOR:VIEW3:FUNCTION VBARS enables the vertical bar type cursors in waveform view 3.
- :CURSOR:FUNCTION? might return :CURSOR:FUNCTION WAVEFORM, indicating that the waveform type cursors are enabled in the default waveform view.

CURSor[:VIEW[x]]:HBArs:POSition[x]

Description
This command sets or queries the position of a horizontal bar cursor. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate.The cursor is specified by x in the :POSition[x] portion of the command, which can be 1 or 2.

Syntax
:CURSor[:VIEW[x]]:HBArs:POSition[x] <NR3>
:CURSor[:VIEW[x]]:HBArs:POSition[x]?

Related Commands
:CURSor[:VIEW[x]]:VBArs:POSition[x]
:CURSor[:VIEW[x]]:HBArs:DELTa?

Arguments
NR3 specifies the cursor position relative to zero for the source waveform.

Returns
The position of the specified horizontal bar cursor.

Examples
- :CURSOR:HBARS:POSITION1 5.0E-6 positions Cursor 1 at 5uW above the zero level of the source waveform in the default waveform view.
- :CURSOR:VIEW2:HBARS:POSITION2? might return :CURSOR:VIEW2:HBARS:POSITION2 -1.68E-6 indicating that in waveform view 2, cursor 2 is 1.68 uW below the zero level of the source waveform.

CURSor[:VIEW[x]]:HBArs:DELTa? (Query Only)

Description
This query only command returns the difference between the two horizontal bar cursors. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate.

Syntax
:CURSor[:VIEW[x]]:HBArs:DELTa?

Related Commands
:CURSor[:VIEW[x]]:VBArs:DELTa?
:CURSor[:VIEW[x]]:HBArs:POSition[x]

Returns

The difference between the two horizontal bar cursors.

Examples

- :CURSOR:VIEW4:HBARS:DELTA? might return :CURSOR:VIEW4:HBARS:DELTA 556.000E-6, indicating a 556uW difference between the two horizontal bar cursors in waveform view 4.

CURSor[:VIEW[x]]:VBArs:POSition[x]

Description
This command sets or queries the position of a vertical bar cursor. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate. The cursor is specified by x in the :POSition[x] potion of the command, which can be 1 or 2.

Syntax
:CURSor[:VIEW[x]]:VBArs:POSition[x] <NR3>
:CURSor[:VIEW[x]]:VBArs:POSition[x]?

Related Commands
:CURSor[:VIEW[x]]:HBArs:POSition[x]
:CURSor[:VIEW[x]]:VBArs:DELTa?

Arguments
NR3 specifies the cursor position measured from the trigger point of the source waveform.

Returns
The position of the specified vertical bar cursor.

Examples

- :CURSOR:VIEW1:VBARS:POSITION1 21E-9 positions Cursor 1 at 21ns from the trigger point of the source waveform in waveform view 1.
- :CURSOR:VBARS:POSITION2? might return :CURSOR:VBARS:POSITION2 3.45E-6 indicating that in the default waveform view, cursor 2 is 3.45us from the trigger point of the source waveform.

CURSor[:VIEW[x]]:VBArs:DELTa? (Query Only)

Description
This query only command returns the difference between the two vertical bar cursors. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate.

Syntax
:CURSor[:VIEW[x]]:VBArs:DELTa?

Related Commands
:CURSor[:VIEW[x]]:HBArs:DELTa?
:CURSor[:VIEW[x]]:VBArs:POSition[x]

Returns
The difference between the two vertical bar cursors.

Examples
:CURSOR:VBARS:DELTA? might return :CURSOR:VBARS:DELTA 3e-12, indicating a 3ps difference between the two vertical bar cursors in the default waveform view.

CURSor[:VIEW[x]]:WAVeform:HPOS[x]? (Query Only)

Description
This query only command returns the position of the specified waveform cursor in vertical units. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate. The cursor is specified by x in the :HPOS[x] portion of the command, which can be 1 or 2.
Because this is waveform cursor mode, this command returns the vertical value in the source waveform which occurs at the time specified by the CURSor:WAVeform:POSition[x] command.

Syntax
:CURSor[:VIEW[x]]:WAVeform:HPOS[x]?

Related Commands
:CURSor[:VIEW[x]]:WAVeform:POSition[x]
:CURSor[:VIEW[x]]:WAVeform:VDELTa?
:CURSor[:VIEW[x]]:WAVeform:HDELTa?

Returns
The position of the specified waveform cursor.

Examples
:CURSOR:VIEW6:WAVEFORM:HPOS2? might return :CURSOR:VIEW6:WAVEFORM:HPOS2 4.67E-4, indicating that in waveform view 6, cursor 2 is at 467uW relative to ground on the source waveform.

CURSor[:VIEW[x]]:WAVeform:POSition[x]

Description
This command sets or queries the position of a waveform cursor in horizontal units (usually time). The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate. The cursor is specified by x in the :POSition[x] portion of the command, which can be 1 or 2.

Syntax
:CURSor[:VIEW[x]]:WAVeform:POSition[x] <NR3>
:CURSor[:VIEW[x]]:WAVeform:POSition[x]?

Related Commands
:CURSor[:VIEW[x]]:WAVeform:HPOS[x]?
:CURSor[:VIEW[x]]:WAVeform:VDELTa?
:CURSor[:VIEW[x]]:WAVeform:HDELTa?

Arguments
<NR3> specifies the cursor position measured relative to the time of the trigger point of the source waveform.

Returns
The position of a waveform cursor.

Examples
- :CURSOR:VIEW2:WAVEFORM:POSITION1 36.8E-9 positions waveform cursor 1 at 36.8ns relative to the time of the trigger point of the source waveform in waveform view 2.
- :CURSOR:WAVEFORM:POSITION2? might return :CURSOR:WAVEFORM:POSITION2 19E-9, indicating that in the default waveform view, waveform cursor 2 is at 19ns relative to the time of the trigger point of the source waveform.

CURSor[:VIEW[x]]:WAVeform:VDELTa?

Description
This query only command returns the vertical difference between the waveform cursors. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate.

Syntax
:CURSor[:VIEW[x]]:WAVeform:VDELTa?

Related Commands
:CURSor[:VIEW[x]]:WAVeform:POSition[x]
:CURSor[:VIEW[x]]:WAVeform:HPOS[x]?
:CURSor[:VIEW[x]]:WAVeform:HDELTa?

Returns
The vertical difference between the waveform cursors.

Examples
:CURSOR:VIEW3:WAVEFORM:VDELTA? might return :CURSOR:VIEW3:WAVEFORM:VDELTA 1.06E-3, indicating that in waveform view 3, the difference between the waveform cursors is 1.06 mW.

CURSor[:VIEW[x]]:WAVeform:HDELTa?

Description
This query only command returns the horizontal difference between the waveform cursors. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate.

Syntax
:CURSor[:VIEW[x]]:WAVeform:HDELTa?

Related Commands
:CURSor[:VIEW[x]]:WAVeform:POSition[x]
:CURSor[:VIEW[x]]:WAVeform:HPOS[x]?
:CURSor[:VIEW[x]]:WAVeform:VDELTa?

Returns
The vertical difference between the waveform cursors.

Examples
:CURSOR:WAVEFORM:HDELTA? might return :CURSOR:WAVEFORM:HDELTA 3.88E-9, indicating that in the default waveform view, the difference between the waveform cursors is 3.88ns.

CURSor[:VIEW[x]]:MODe

Description
This command sets or queries the cursor mode. The optional [:VIEW[x]] argument specifies which waveform view's cursors to manipulate.

Syntax
:CURSor[:VIEW[x]]:MODe { INDependent | LINKed }
:CURSor[:VIEW[x]]:MODe?

Arguments

- INDependent sets the cursor mode to independent, where moving one cursor does not move the other.
- LINKed set the cursor mode to linked, where TSOVu does its best to maintain the same delta between the two cursors while one is moved.

Returns
The current cursor mode.

Examples
- :CURSOR:VIEW1:MODE LINKED sets the current cursor mode to linked in waveform view 1.
- :CURSOR:MODE? might return :CURSOR:MODE INDEPENDENT, indicating that in the default waveform view, the current cursor mode is independent.

CURSor[:VIEW[x]]:WFMSource

Description
This command sets or queries the Source Waveform mode. Source Waveform mode defines whether the set of cursors share a waveform source or can have split waveform sources.

Syntax
:CURSor[:VIEW[x]]:WFMSource { SAMe | SPLit }
:CURSor[:VIEW[x]]:WFMSource?

Related Commands
:CURSor[:VIEW[x]]:CURSOR[x]:SOUrce

Arguments
- SAMe sets the Source Waveform mode to Same, meaning all cursors will have the same waveform source.
- SPLit sets the Source Waveform mode to Split, meaning each cursor can have a different waveform source.

Returns
The Source Waveform mode.

Examples
- :CURSOR:WFMSOURCE SAME sets the Source Waveform mode to Same in the default waveform view.
- :CURSOR:VIEW2:WFMSOURCE? might return :CURSOR:VIEW2:WFMSOURCE SPLIT, indicating that in waveform view 2, the Source Waveform mode is set to Split.

Diagnostic

DIAG:POWERUP:STATUS?

Description

This is a query only command that returns a result of the power on diagnostic execution. "Pass" indicates the system has passed the diagnostic test, it is similar for "Fail" case.

Syntax
DIAG:POWERUP:STATUS?

Returns

<QString>

Examples
DIAG:POWERUP:STATUS? might return DIAG:POWERUP:STATUS? "PASS", indicating that the power on diagnostic was passed.

Display Control Command Group

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

You can set the following:
- Histogram
- Whether cursor and histogram are displayed.
- Whether waveforms are displayed (shown) or not displayed (hidden).
- Whether waveforms are displayed in Normal mode as dots or vectors, in
- Variable Persistence mode, or in Infinite Persistence mode.
- If interpolation is used, which type (Sin(x) or Linear).
- The style of graticule that underlies the waveforms.

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

DISplay:MODE

Description

This command will get or set the mode of the display view.

Syntax
DISplay:MODE { OVERlay | TILE }
DISplay:MODE?

Arguments
- OVERlay
- TILE

Examples
DISplay:MODE TILE sets thedisplay mode to TILE.
DISplay:MODE? might return DISplay:MODE TILE if TILE is selected.

DISplay:WAVeform:VIEW[x]:GRATicule:STYLe

Description
This command will get or set the style of graticule that is displayed.
The display view is specified by x.

Syntax
DISplay:WAVeform:VIEW[x]:GRATicule:STYLe { TIME|FULL | NONE | GRID }
DISplay:WAVeform:VIEW[x]:GRATicule:STYLe?

Arguments
- FULL specifies a frame and a grid.
- TIME specifies vertical grid related to time
- GRID specifies a frame and a grid.
- NONE means no grid at all.

Examples

- DISplay:WAVeform:VIEW1:GRATicule:STYLe GRID sets the graticule style to display a frame and a grid which is on display view 1.
- DISplay:WAVeform:VIEW1:GRATicule:STYLe? might return
- DISplay:WAVeform:VIEW1:GRATicule:STYLe FULL when all graticule elements (grid and frame) are displayed which is on display view 1.

DISplay:WAVeform:VIEW[x]:GRATicule:INTensity

Description
This command will get or set the graticule intensity of the specified display view.
The display view is specified by x.

Syntax
DISplay:WAVeform:VIEW[x]:GRATicule:INTensity <NR3>
DISplay:WAVeform:VIEW[x]:GRATicule:INTensity?

Arguments
<NR3> is the graticule intensity of the waveform in percentage.

Returns
It returns the graticule intensity of the reference waveform or live waveform of specified display view.

Examples

- DISplay:WAVeform:VIEW1:GRATicule:INTensity 70 sets the graticule intensity 70 percentage which is on display view 1
- DISplay:WAVeform:VIEW1:GRATicule:INTensity?
  Might return DISplay:WAVeform:VIEW1:GRATicule:INTensity 70, indication that the graticlue display is 70 percentage on display view 1.

DISplay:WAVeform:VIEW[x]:WINTensity

Description
This command will get or set the waveform intensity of the waveform of specified display view.
The display view is specified by x.

Syntax
DISplay:WAVeform:VIEW[x]:WINTensity<NR3>
DISplay:WAVeform:VIEW[x]:WINTensity?

Arguments
<NR3> is the waveform intensity of the waveform in percentage.

Returns
It returns the waveform intensity of the reference waveform or live waveform of specified display view.

Examples

- DISplay:WAVeform:VIEW1:WINTensity 70 sets the waveform intensity 70 percentage which is on display view 1
- DISplay:WAVeform:VIEW1:WINTensity? Might return DISplay:WAVeform:VIEW1:WINTensity 70, indication that the waveform display is 70 percentage on display view 1.

DISplay:WAVeform:VIEW[x]:WIPolate

Description
This command will get or set the interpolation algorithm used to display any waveform.
The display view is specified by x.

Syntax
DISplay:WAVeform:VIEW[x]:WIPolate { SINX | LINear | NONE }
DISplay:WAVeform:VIEW[x]:WIPolate?

Arguments

- SINX specifies Sin (x)/x interpolation. This algorithm computes points using a curve fit between the actual values acquired. It assumes all interpolated points fall along the curve. This is useful when displaying more rounded waveforms such as sine waves. This algorithm can be used for general use, but it may introduce some overshoot or undershoot in signals with fast rise times
- LINear specifies linear interpolation. This algorithm computes points between actual acquired samples by using a straight line fit. The algorithm assumes all interpolated points fall along the straight line. Linear interpolation is useful for many waveforms such as pulse trains.
- NONE turns off the interpolation function.

Returns
It returns the interpolation method in query form.

Limitation
This command only applies to pattern mode.
When Waveform Style is selected as DOTS Interpolation dropdown will contain Sin (x)/x, Linear, None.
When Waveform Style is selected as VECTORS Interpolation dropdown will contain Sin (x)/x, Linear.

Examples
DISplay:WAVeform:VIEW2:WIPolate LINEAR selects the linear interpolation algorithm which is on display view 1.
DISplay:WAVeform:VIEW2:WIPolate? might return DISplay:WAVeform:VIEW2:WIPolate LINEAR, indicating that linear interpolation algorithm is selected which is on display view 2.

DISplay:WAVeform:VIEW[x]:ZOOM:STATe

Description
This command will get or set the state of Zoom of the specified display view.
The display view is specified by x.

Syntax
DISplay:WAVeform:VIEW[x]:ZOOM:STATe { OFF | ON }
DISplay:WAVeform:VIEW[x]:ZOOM:STATe?

Arguments
- OFF
- ON

Examples
- DISplay:WAVeform:VIEW1:ZOOM:STATe OFF switch off the zoom for display view 1.
- DISplay:WAVeform:VIEW1:ZOOM:STATe? might return DISplay:WAVeform:VIEW[x]:ZOOM:STATe ON if zoom is on for display view 1.

DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:POSition

Description
This command will get or set the Horizontal position of zoom for specified display view.
The display view is specified by x..

Syntax
DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:POSition <NR3>
DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:POSition?

Arguments
<NR3> is the Horizontal position of the zoom.

Returns
It returns the Horizontal position of the zoom window of specified display view.

Examples
- DISplay:WAVeform:VIEW1:ZOOM:HORizontal:POSition 70 sets the horizontal position to 70 which is on display view 1
- DISplay:WAVeform:VIEW1:ZOOM:HORizontal:POSition?
  Might return DISplay:WAVeform:VIEW1:ZOOM:HORizontal:POSition 70, indication that the horizontal position is 70 on display view 1.

DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:SCALe

Description
This command will get or set the Horizontal scale of zoom for specified display view.
The display view is specified by x.

Syntax
DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:SCALe <NR3>
DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:SCALe?

Arguments
<NR3> is the Horizontal scale of the zoom.

Returns
It returns the Horizontal scale of the zoom window of specified display view.

Examples
- DISplay:WAVeform:VIEW1:ZOOM:HORizontal:SCALe 3 sets the horizontal scale to 3 which is on display view 1
- DISplay:WAVeform:VIEW1:ZOOM:HORizontal:SCALe?
  Might return DISplay:WAVeform:VIEW1:ZOOM:HORizontal:SCALe 5, indication that the horizontal scale is 5 on display view 1.

DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:WINScale

Description
This command will get or set the Horizontal scale in window of zoom for specified display view.
The display view is specified by x.

Syntax

DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:WINScale <NR3>
DISplay:WAVeform:VIEW[x]:ZOOM:HORizontal:WINScale?

Arguments
<NR3> is the Horizontal scale in window of the zoom.

Returns
It returns the Horizontal scale in window of the zoom window of specified display view.

Examples

- DISplay:WAVeform:VIEW1:ZOOM:HORizontal:WINScale 2e-12 sets the horizontal scale in window of zoom to 2ps/div which is on display view 1
- DISplay:WAVeform:VIEW1:ZOOM:HORizontal:WINScale? Might return DISplay:WAVeform:VIEW1:ZOOM:HORizontal:WINScale 2E-9, which is on display view 1.

DISplay:ERRor:DIALog

Description
This command enables or disables error dialogs from displaying on the UI when an error condition occurs.

Syntax
DISplay:ERRor:DIALog {ON | OFF | 1 | 0}
DISplay:ERRor:DIALog?

Arguments
0 or OFF hides the error dialogs.
1 or ON displays the error dialogs.

Returns
The query version of this command returns 1 or 0.

Examples
DISPLAY:ERROR:DIALOG 0 hides the error dialogs from display.
DISPLAY:ERROR:DIALOG? might return 1, indicating that error messages will be displayed on the main window.

DISplay:REF[x]

Description

The user shall use this PI command to set or query whether the specified reference waveform is displayed.
The waveform is specified by x. This is equivalent to the Display toggle that is available in the Ref Configuration Menu (right-click property of the Ref Badge in the Settings Bar at the bottom of the user interface).
NOTE: You should define a reference waveform before turning the waveform on.
Group: Vertical

Syntax
DISplay:REF[x] { ON | OFF | 0 | 1 }
DISplay:REF[x]?

Arguments
1. ON displays the specified reference waveform.
2. OFF turns off the display of the specified reference waveform.
3. NR1 set to 0 turns off the display of the specified reference waveform; any other value displays the specified reference waveform.

Returns
It returns the display status of the waveform.

Examples
DISPLAY:REF1 0: it turns off the REF1 display.
DISPLAY:REF1?: it returns the DISPLAY:REF1 0, the waveform display status.

DISplay:M[n]{A|B}

Description
The user shall use this PI command to set or query whether the specified live waveform is displayed. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
DISplay:M[n]{A|B} { ON | OFF | 0 | 1 }
DISplay:M[n]{A|B}?

Arguments
1. { ON | OFF | 0 | 1 }: ON or 1 displays the specified live waveform. OFF or 0 turns off the display of the specified live waveform.

Returns
It returns the status of the waveform display.

Examples
DISPLAY:M1A ON displays M1A waveform.
DISPLAY:M1A? might return 0 to signify that the M1A waveform is not currently being displayed.

Histogram Command Group

Histogram commands lets you select the type of histogram, what part of the waveform should go into the histogram, and histogram statistics.

You can use commands from this group to do the following:
- Select any channel or reference waveform and create a histogram of vertical or horizontal values for it.
- Adjust the limits of the box that define the area on the waveform from which the histogram data is obtained. The histogram box can be set using source waveform coordinates or percentage-of-display coordinates.
- Create a linear or logarithmic plot of histogram data and set plot size and color.
- Turn the display of the histogram on or off.
- Set or query the color of the histogram box and histogram plot.
- Get histogram statistics, such as total hits, mean value, peak-to-peak value, and standard deviation.
- Get all the histogram parameters

:HISTogram:ADDHisto

Description
This command adds a histogram using the specified source, mode, area type, left, top, right and bottom boundary limits.

Syntax
HISTogram:ADDHisto <source>, {HORizontal | VERTical}, {ABSolute | PERCentage}, <NR3>,<NR3>,<NR3>,<NR3>

Arguments
<source> can be any of the following three:

1. M[x]A|B selects a channel waveform as the source or destination waveform
2. MATH<x> selects a math waveform as the source for the histogram
3. REF<x> selects a reference waveform as the source for the histogram

HORizontal creates a horizontally positioned histogram that shows time distribution
VERTical creates a vertically positioned histogram that shows a voltage distribution (or other vertical distribution, such as amperes)

ABSolute specifies that the histogram plot box boundary limits are specified in absolute values
PERCentage specifies that the histogram plot box boundary limits are specified in percentage values

<NR3> (first) is the left position of the histogram box
<NR3> (second) is the top position of the histogram box
<NR3> (third) is the right position of the histogram box
<NR3> (fourth) is the bottom position of the histogram box

Examples
HISTOGRAM:ADDHISTO REF1,VERTICAL,ABSOLUTE,20.5E-9,248.9E-3,22.5E-9,-251.1E-3 adds a vertical Histogram1 with source as Ref1 and whose boundary limits are specified in terms of absolute values.

:HISTogram:DELete:ALL

Description
This command deletes all active histograms.

Syntax
HISTogram:DELete:ALL

Examples
HISTOGRAM:DELETE:ALL will delete all active histograms

:HISTogram:HISTo<x>:CONFig:DISPlay

Description
This command sets or queries the display setting for the given histogram. Histograms are specified by <x>. This command is used to add/remove the histogram plot to the display associated with it's source.

Syntax
HISTogram:HISTo<x>:CONFig:DISPlay { ON | OFF | 0 | 1 }

Arguments

ON or any other non-zero value adds the histogram plot to the display associated with it's source
OFF or 0 removes the histogram plot from the display associated with it's source

Returns
0 or 1 indicating state of display of the specified histogram is OFF or ON respectively

Examples

- HISTOGRAM:HISTO1:CONFIG:DISPLAY ON adds the HISTO1's plot to the display associated with its source.
- HISTOGRAM:HISTO2:CONFIG:DISPLAY OFF removes the HISTO2's plot from the display associated with it's source
- HISTOGRAM:HISTO3:CONFIG:DISPLAY 1 adds the HISTO3's plot to the display associated with it's source
- HISTOGRAM:HISTO1:CONFIG:DISPLAY? might return HISTOGRAM:HISTO1:CONFIG:DISPLAY 1 meaning the HISTO1's plot is added to the display associated with it's source

:HISTogram:HISTo<x>:CONFig:SOURce

Description
This command sets or queries the source of the histogram measurement. Histograms are specified by <x>. The waveform need not be displayed for histograms to run.

Syntax
:HISTogram:HISTo<x>:CONFig:SOURce <source>

Arguments
<source> can be any one of the following:
M[x]A|B selects a channel waveform as the source or destination waveform.
MATH<x> selects a math waveform as the source for the histogram
REF<x> selects a reference waveform as the source for the histogram

Returns
The source of the specified histogram

Examples

- HISTOGRAM:HISTO1:CONFIG:SOURCE REF2 sets REF2 as the source waveform for Histogram1
- HISTOGRAM:HISTO2:CONFIG:SOURCE M1B sets channel B on module 1 as the source waveform for Histogram2
- HISTOGRAM:HISTO1:CONFIG:SOURCE? Might return HISTOGRAM:HISTO1:CONFIG:SOURCE REF2 indicating that the waveform source for Histogram1 is Ref1

:HISTogram:HISTo<x>:CONFig:MODe

Description
This command sets or queries the mode of the given histogram. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:CONFig:MODe {HORizontal | VERTical}

Arguments

HORizontal creates a horizontally positioned histogram that shows time distribution
VERTical creates a vertically positioned histogram that shows a voltage distribution (or other vertical distribution, such as amperes)

Returns
HORIZONTAL indicating the histogram is horizontally positioned showing time distribution
VERTICAL indicating the histogram is vertically positioned showing voltage distribution (or other vertical distribution, such as amperes)

Examples
HISTOGRAM:HISTO1:CONFIG:MODE HORIZONTAL configures Histogram1 to be horizontally positioned
HISTOGRAM:HISTO2:CONFIG:MODE? might return
HISTOGRAM:HISTO2:CONFIG:MODE VERTICAL indicating Histogram2 is vertically positioned

:HISTogram:HISTo<x>:CONFig:TYPE

Description
This command sets or queries whether the histogram is calculated linearly or logarithmically. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:CONFig:TYPE {LINear | LOG }

Arguments
LINear specifies that bin counts smaller than the maximum should be scaled linearly by dividing the bin count by the maximum bin count.
LOG specifies that bin counts smaller than the maximum should be scaled logarithmically (log (bin-count)) with log(0) staying at 0 (baseline). The base of the log does not matter since logs to different bases differ only by a constant multiplier. Logarithmic scaling provides better visual detail for bins with low counts.

Returns
LINEAR indicating the histogram is displayed linearly.
LOG indicating the histogram is displayed logarithmically.

Examples
HISTOGRAM:HISTO1:CONFIG:TYPE LINEAR displays the count in each bin scaled linearly
HISTOGRAM:HISTO2:CONFIG:TYPE? might return
HISTOGRAM:HISTO2:CONFIG:TYPE LINEAR, indicating that the histogram display is scaled linearly

:HISTogram:HISTo<x>:CONFig:AREA

Description
This command sets or queries whether boundary limits of histogram plot box is specified in absolute or percentage.

Syntax
HISTogram:HISTo<x>:CONFig:AREA {ABSolute | PERCentage}

Arguments
ABSolute specifies that the histogram plot box boundary limits are specified in absolute values

PERCentage specifies that the histogram plot box boundary limits are specified in percentage values

Returns
ABSOLUTE indicating that the histogram plot box boundary limits are specified in absolute values
PERCENTAGE indicating that the histogram plot box boundary limits are specified in percentage values

Examples
HISTOGRAM:HISTO1:CONFIG:AREA ABSOLUTE sets the boundary limits of HISTO1's plot box to be specified in absolute
HISTOGRAM:HISTO2:CONFIG:AREA? might return
HISTOGRAM:HISTO2:CONFIG:AREA PERCENTAGE, indicating that the HISTO2's plot box is specified in percentage

:HISTogram:HISTo<x>:CONFig:BOX

Description
This command sets or queries the left, top, right, and bottom boundaries of the histogram box in source waveform coordinates (absolute values). Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:CONFig:BOX <NR3>,<NR3>,<NR3>,<NR3>

Arguments
<NR3> (first) is the left position of the histogram box in source waveform coordinates
<NR3> (second) is the top position of the histogram box in source waveform coordinates
<NR3> (third) is the right position of the histogram box in source waveform coordinates
<NR3> (fourth) is the bottom position of the histogram box in source waveform coordinates

Returns
A comma separated list of left, top, right and bottom positions of histogram box in source waveform coordinates

Examples
- HISTOGRAM:HISTO1:CONFIG:BOX 1.518E-006,-2.46E-1,3.518E-6,-7.47E-1 defines the coordinates of the HISTO1's histogram box in source waveform coordinates. HISTOGRAM:HISTO2:CONFIG:BOX? might return
- HISTOGRAM:HISTO2:BOX 1.51800000000E-006,-0.246000000000,3.51800000000E-006, 0.747000000000 indicating the left, top, right and bottom positions of HISTO2's box in source waveform coordinates respectively.

:HISTogram:HISTo<x>:STATistics:HITS (Query only)

Description

This command is used to get the total hits calculated for the specified histogram. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:STATistics:HITS?

Arguments
This query only command shall have no arguments

Returns
<NR3> the hits value for the specified histogram

Examples
HISTOGRAM:HISTO1:STATISTICS:HITS? might return
HISTOGRAM:HISTO1:STATISTICS:HITS 6.83400000000E+003, indicating that the
total hits for Histogram1 is 6,834

:HISTogram:HISTo<x>:STATistics:MEAN (Query only)

Description
This command is used to get the mean value calculated for the specified histogram. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:STATistics:MEAN?

Arguments
This query only command shall have no arguments

Returns
<NR3> the mean value for the specified histogram

Examples
HISTOGRAM:HISTO2:STATISTICS:MEAN? might return
HISTOGRAM:HISTO2:STATISTICS:MEAN 43.0000000000E–009, indicating that the
mean value for Histogram2 is 43 ns

:HISTogram:HISTo<x>:STATistics:MEDian (Query only)

Description
This command is used to get the median value calculated for the specified histogram. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:STATistics:MEDian?

Arguments
This query only command shall have no arguments

Returns
<NR3> the median value for the specified histogram

Examples
HISTOGRAM:HISTO1:STATISTICS:MEDIAN? might return
HISTOGRAM:HISTO1:STATISTICS:MEDIAN 43.0000000000E–009, indicating that the
median value for Histogram1 is 43 ns

:HISTogram:HISTo<x>:STATistics:MODe (Query only)

Description
This command is used to get the bin having maximum hits for the specified histogram. Histograms are specified by <x>.

Syntax

HISTogram:HISTo<x>:STATistics:MODe?

Arguments
This query only command shall have no arguments

Returns
<NR3> the bin having maximum hits for the specified histogram

Examples
HISTOGRAM:HISTO3:STATISTICS:MODE? might return
HISTOGRAM:HISTO3:STATISTICS:MODE 390.0000000000E-6, indicating that the bin having maximum hits value for the waveform source of Histogram3 is 390µ

:HISTogram:HISTo<x>:STATistics:PKTopk (Query only)

Description
This command is used to get the peak-to-peak value calculated for the specified histogram. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:STATistics:PKTopk?

Arguments
This query only command shall have no arguments

Returns
<NR3> the peak-to-peak value for the specified histogram

Examples
HISTOGRAM:HISTO1:STATISTICS:PKTOPK? might return
HISTOGRAM:HISTO1:STATISTICS:PKTOPK 20.0000000000E–009, indicating that the peak-to-peak value for Histogram1 is 20 ns

:HISTogram:HISTo<x>:STATistics:STDDev (Query only)

Description
This command is used to get the standard deviation value calculated for the specified histogram. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:STATistics:STDDev?

Arguments
Query only command shall have no arguments

Returns
<NR3> the standard deviation value for the specified histogram

Examples
HISTOGRAM:HISTO4:STATISTICS:STDDEV? might return
HISTOGRAM:HISTO4:STATISTICS:STDDEV 5.80230767128E–009, indicating that the standard deviation value for Histogram4 is 5.80 ns

:HISTogram:HISTo<x>:STATistics:WAVeforms (Query only)

Description
This command is used to get the number of waveforms used in the specified histogram. Histograms are specified by <x>.

Syntax
HISTogram:HISTo<x>:STATistics:WAVeforms?

Arguments
This query only command shall have no arguments

Returns
<NR3> the number of waveforms used in the specified histogram

Examples
HISTOGRAM:HISTO1:STATISTICS:WAVEFORMS? might return
HISTOGRAM:HISTO1:STATISTICS:WAVEFORMS 2.08100000000E+003, indicating that 2081 waveforms were used to create Histogram1

:HISTogram:HISTo<x>:DELete

Description
This command is used to delete the specified histogram.

Syntax
HISTogram:HISTo<x>:DELete

Examples
HISTOGRAM:HISTO3:DELETE will delete Histogram3

Horizontal Command Group

You use the commands from the Horizontal Command Group to control the time bases of the instrument.

HORizontal:APOSition

Description
This command sets or queries auto position. Setting auto position is available only in eye mode. Enabling auto position mode will center the eye on the display.

Syntax
:HORizontal:APOSition {ON | OFF | 1 | 0}
:HORizontal:APOSition?

Arguments
ON or 1 turns on auto position.
OFF or 0 turns off auto position.

Returns

The query version of this command returns 1 or 0.

Examples

- :HORIZONTAL:APOSition ON sets Auto Position to On.
- :HORIZONTAL:APOSition? might return 0, indicating that Auto Position is Off.

Limitations

This command is set/query only when Pattern Sync is Off.

HORizontal[:MAIN]:REFPoint

Description

This command sets or queries the horizontal reference point in percentage.

The horizontal reference point is the point that holds stationary when horizontal scale changes. The only time this rule is broken, is when it would cause the acquisition window to extend beyond the beginning or end of a pattern.

Syntax

:HORizontal[:MAIN]:REFPoint <NR3>
:HORizontal[:MAIN]:REFPoint?

Related Commands

:HORizontal[:MAIN]:POSition

Arguments

<NR3> is the percentage of the record at which the horizontal reference is set. The range is 0 through 100 (corresponding to 0% through 100% of the record.)

Returns

The query version of this command returns an NR3 value between 0 and 100, representing the fraction of the record at which the horizontal reference point is set.

Examples

- :HORIZONTAL:REFPOINT 25 sets the horizontal reference point to 25% of Record Length.
- :HORIZONTAL:REFPOINT? might return ":HORIZONTAL:REFPOINT 25.0000000000".

:HORizontal[:MAIN]:POSition

Description

This command sets or queries the horizontal position in seconds.

The horizontal position is the time between the trigger and the first acquired point in a record.

Syntax

:HORizontal[:MAIN]:POSition <NR3>
:HORizontal[:MAIN]:POSition?

Related Commands

:HORizontal[:MAIN]:REFPoint

Arguments

<NR3> is the horizontal position in seconds. The valid range is defined by the instrument that TSOVu is connected to.

Returns
The query version of this command returns an NR3 value representing the horizontal position in seconds.

Examples
- :HORIZONTAL:POSITION 30e-9 sets the horizontal position be 30 ns.
- :HORIZONTAL:POSITION? might return ":HORIZONTAL:POSITION 30.0000000000E-9" indicating that horizontal position is set to 30ns.

:HORizontal[:MAIN]:SCALe

Description
This command sets or queries the horizontal scale (time per division).

Syntax
:HORizontal[:MAIN]:SCALe <NR3>
:HORizontal[:MAIN]:SCALe?

Related Commands
:HORizontal[:MAIN]:RESolution?
:HORizontal:PLENgth
:HORizontal:SRATe

Arguments
<NR3> is the horizontal time per division in seconds.

Returns
The query version of this command returns an NR3 value for the horizontal scale value is seconds.

Examples
- :HORIZONTAL:SCALE 2.5E-9 sets the horizontal scale to 2.5ns per division.
- :HORIZONTAL:SCALE? might return ":HORIZONTAL:SCALE 2.50000000000E-9"

Limitations
This command is query only when Full Pattern is On and Pattern Sync is On.

:HORizontal[:MAIN]:RECordlength

Description
This command sets or queries the record length in samples.

Syntax
:HORizontal[:MAIN]:RECordlength <NR1>

Related Commands
:HORizontal:PLENgth
:HORizontal:SAMPlesui

Arguments
<NR1> is the integer value of the record length in samples. The valid range is defined by the instrument that TSOVu is connected to.

Returns
The query version of this command returns an NR1 value of the record length.

Examples
:HORIZONTAL:RECordlength 1e+4 sets the record length to be 10000.
:HORIZONTAL:RECordlength? might return ":HORIZONTAL:RECORDLENGTH 10.0000000000E+3" as the record length value.

Limitations
This command is query only when Full Pattern is On and Pattern Sync is On.

:HORizontal[:MAIN]:RESolution (Query only)

Description
This command returns the current resolution per sample in seconds, which is the time between two samples.

Syntax
:HORizontal[:MAIN]:RESolution?

Related Commands
:HORizontal[:MAIN]:SCALe
:HORizontal:SAMPlesui
:HORizontal[:MAIN]:SCALe
:HORizontal:SRATe
:HORizontal[:MAIN]:RECordlength

Returns
This query command returns an NR3 value representing the time between any two samples in seconds.

Examples
:HORIZONTAL:RESolution? might return ":HORIZONTAL:RESolution 1.9820606061E-12",
which indicates the horizontal resolution is 1.982ps.

HORizontal:SAMPlesui

Description
This command sets or queries the samples per UI.

Syntax
HORizontal:SAMPlesUI <NR1>
HORizontal:SAMPlesUI?

Related Commands
HORizontal:SAMPlesUI <NR1>

Arguments
<NR1> is the integer value that sets the value of samples per UI.

Returns
The query version of this command returns an NR1 value as the samples per UI.

Examples
- HORIZONTAL:SAMPlesui 20 sets the samples per UI to be 20.

- HORIZONTAL:SAMPlesui? might return "HORIZONTAL:SAMPLESUI 20.0000000000"

Limitations
This command is query only when Pattern Sync is Off.

HORizontal:PLENgth

Description
Set or query the number of symbols in a pattern.

Syntax
HORizontal:PLENgth <NR1>

Related Commands
TRIGger:PSYNc:PLENgth

Arguments
<NR1> is the integer value of number of symbols in a pattern.

Returns
The query version of this command returns an NR1 value as the number of symbols in a pattern.

Examples
HORIZONTAL:PLENgth 32760 sets the pattern length to be 32760, and the Pattern name becomes "User Defined" in UI.
HORIZONTAL:PLENgth? might return "HORIZONTAL:PLENgth 32.7600000000E+3"

HORizontal:SRATe

Description
This command sets or queries the symbol rate, which is equivalent to the signal baud rate.

Syntax
HORizontal:SRATe <NR3>

Related Commands
TRIGger:PSYNc:DATARate
HORizontal[:MAIN]:SCALe

Arguments
<NR3> is the value of the symbol rate. The valid range is defined by the instrument TSOVu is connected to.

Returns
The query version of this command returns an NR3 value of the symbol rate.

Examples
HORIZONTAL:SRATE 2.5E+9 sets the symbol rate to be 2.5 G.
HORIZONTAL:SRATE? might return HORIZONTAL:SRATE 2.5000000000E+9

HORizontal:PSYNc

Description

This command sets or queries pattern sync. Setting pattern sync to Off puts the instrument in eye mode.

Syntax
HORizontal:PSYNc {ON | OFF | 1 | 0}

Related Commands
HORizontal:FPATtern

Arguments
ON or 1 turns on pattern sync
OFF or 0 turns off pattern sync

Returns
The query version of this command returns 1 or 0.

Examples
HORIZONTAL:PSYNC ON sets Pattern Sync to be on.
HORIZONTAL:PSYNC? might return HORIZONTAL:PSYNC 0 indicating that pattern sync is Off.

HORizontal:DCRAtio

Description
This command sets or queries the data-to-clock ratio (<data rate>,<clock rate>). The first <NR1> value represents the data rate and the second <NR1> value represents the clock rate.

Syntax
:HORizontal:DCRAtio <NR1>,<NR1>

Related Commands
:TRIGger:PSYNc:DCRAtio

Arguments
<NR1> (first argument) sets the data rate.
<NR1> (second argument) sets the clock rate.

The valid Data Rate:Clock Rate ratios are
1:1
2:1
4:1
8:1
16:1
32:1

Returns
The query version of this command returns two commas separated NR1 values, the first being Data Rate and the second being Clock Rate.

Examples
- :HORIZONTAL:DCRATIO 2,1 sets the data-to-clock ratio as 2:1.
- :HORIZONTAL:DCRATIO? might return :HORIZONTAL:DCRATIO 16,1 indicating a data-to-clock ratio of 16:1.

:HORizontal:REF<x>[:MAIN]:RESolution? (Query Only)

Description

This query only command returns the current resolution per sample of the reference waveform.

Syntax
:HORizontal:REF<x>[:MAIN]:RESolution?

Related Commands
:HORizontal:REF<x>[:MAIN]:RECordlength?
:HORizontal:REF<x>[:MAIN]:SCALe?

Returns
This query returns an NR3 value representing the current resolution per sample of the reference waveform.

Examples
:HORIZONTAL:REF1:RESOLUTION? might return ":HORIZONTAL:REF1:RESOLUTION 16.6666668892E-12", indicating 16.667ps between each sample in the reference waveform.

:HORizontal:REF<x>[:MAIN]:RECordlength? (Query Only)

Description
This query only command returns the record length of the reference waveform.

Syntax
:HORizontal:REF<x>[:MAIN]:RECordlength?

Related Commands
:HORizontal:REF<x>[:MAIN]:RESolution?
:HORizontal:REF<x>[:MAIN]:SCALe?

Returns
The NR3 value representing the record length of the reference waveform.

Examples
:HORIZONTAL:REF1:RECORDLENGTH? might return ":HORIZONTAL:REF1:RECORDLENGTH 327.6400000000E+3" indicating a record length of 327,640 samples.

:HORizontal:REF<x>[:MAIN]:SCALe? (Query Only)

Description
This query only command returns the horizontal scale (time per division) of the specified reference waveform.

Syntax
:HORizontal:REF<x>[:MAIN]:SCALe?

Related Commands
:HORizontal:REF<x>[:MAIN]:RESolution?
:HORizontal:REF<x>[:MAIN]:RECordlength?

Returns
The NR3 value representing the horizontal scale of the specified reference waveform.

Examples

:HORIZONTAL:REF1:SCALE? might return ":HORIZONTAL:REF1:SCALE 15.4318439998E-9" indicating 15.43ns per division for the horizontal scale.

:HORizontal:REF<x>[:MAIN]:TOFPoint? (Query Only)

Description
This query only command returns time of first point of the specified reference waveform.

Syntax
:HORizontal:REF<x>[:MAIN]:TOFPoint?

Returns
The NR3 value representing the time of first point of the reference waveform.

Examples
:HORIZONTAL:REF3:TOFPOINT? might return ":HORIZONTAL:REF1:TOFPOINT 0.0000" indicating a time of first point of 0s.

Limitations
If the reference waveform specified is a waveform database, this command will return the IEEE standard value for Not a Number.

Licensing Command Group

LICense:COUNt?

Description
This query returns a count of the number of active licenses installed.

Syntax
LICense:COUNt?

Returns
A count of the number of active licenses installed.

Examples
LICENSE:COUNT? might return :LICENSE:COUNT 2 indicating that 2 active licenses are installed.

LICense:APPID?

Description
This query returns a comma-separated list of the active application IDs.

Syntax
LICENSE:APPID?

Returns
This query returns a comma-separated list of the active application IDs.

Examples
LIC:APPID? might return :LICENSE:APPID "NRZ,PAM4," which is a complete list of the active applications.

LICense:ITEM?

Description
This query returns the nomenclatures, type, descriptions, checked out date, License ID to a specific license. The NR1 argument is zero-indexed. If no argument is provided, zero is assumed.

Syntax
LICense:ITEM? <NR1>

Arguments
<NR1> is the zero-indexed argument specifying a specific license.

Returns
This query returns the nomenclatures, type, descriptions, checked out date, License ID to a specific license.

Examples
LICENSE:ITEM? 1 might return LICENSE:ITEM0 "TSO8SW-NL1-NRZ,Fixed,2/4/2020 9:15:43 AM,949667294,""NRZ"",""ENGINEERING LICENSE - License; NRZ Optical Measurements; Node-Locked 1-Year Subscription"""

LICense:LIST?

Description
This query returns the active license nomenclatures as a comma-separated list of strings. Duplicate nomenclatures, that is, the same license but with different expiration dates, are included.

Syntax
LICense:LIST?

Returns
The active license nomenclatures as a comma-separated list of strings.

Examples
LICENSE:LIST? might return :LICENSE:LIST "TSO8SW-FL1-PAM4-O,Floating,ENGINEERING LICENSE - License; PAM4 Optical Measurements; Floating 1-Year Subscription, TSO8SW-NL1-PAM4-O,Fixed,ENGINEERING LICENSE - License; PAM4 Optical Measurements; Node-Locked 1-Year Subscription"

LICense:HID?

Description
This query returns the TSOVu HostID unique identifier.

Syntax
LICense:HID?

Returns
The TSOVu HostID unique identifier.

Examples
LICENSE:HID? might return LICENSE:HID "TSO-JVSCGZBGK4PJYKH5"

LICense:INSTall:FILE

Description
This command accepts a <File_Path> string with license path and installs it on the instrument.

Syntax
LICense:INSTall:FILE "<File_Path>"

Arguments
<File_Path> is the license file name with path.

Examples
:LICense:INSTall:FILE "C:\Users\sacb\Documents\License\-_-_TSO-B4SSD2AHTU2AFPFL_TSO8SW-NLP-PAM4-O_ENTER (1).LIC”

LICense:INSTall:OPTion

Description
This command sets or queries the current selection for license installation. If TSOVu is the selected option, then the license shall be installed on the host application. If INSTrument is the selected option, then the license shall be installed on the connected instrument.

Syntax
LICense:INSTall:OPTion {TSOVu | INSTrument}
LICense:INSTall:OPTion?

Arguments
TSOVu sets the license to be installed on the host application.
INSTrument sets the license to be installed on the connected instrument.

Returns
TSOVu means the license is set to be installed on the host application.
INSTrument means the license is set to be installed on the connected instrument

Examples
LICense:INSTall:OPTion? might return TSOVU, indicating that the license will be installed on the host application, which is TSOVu.
LICense:INSTall:OPTion INSTrument might set the current install option to connected instrument and any subsequent license installation shall be done on that connected instrument.

LICense:UNINSTALL?

Description
Returns the exit license indicated for the user to return to their TekAMS account. License ID can be used to specify uninstalled license. the exit-license is returned as block-data.

Syntax
LICense:UNINSTALL? "<License ID>"

Arguments
<License ID> License ID of the installed license.

Returns

The exit-license is returned as block-data.

Examples
LIC:UNINSTALL? “569765772” uninstalls the license with the given license ID and returns the license block data.

Measurement Command Group

:MEASUrement:ADDMEAS

:MEASUrement:ADDMEAS <string_category>,<string_name>,<source1>[,<source2>] [,<meas[x]>]

Description
This command adds a measurement from the specified category on the given source or sources with specified measurement ID.

Syntax
:MEASUrement:ADDMeas <category>, <measType>, { M[n]{A|B} | MATH[x] | REF[x] } [, { M[n]{A|B} | MATH[x] | REF[x] }] [, MEAS[x]]

Arguments
<category> is the name of the measurement group
<measType> is the type of an available measurement as a quoted string

{ M[n]{A|B} | MATH[x] | REF[x] } is the primary source for the measurement:

- M[n]{A|B} selects a channel waveform source.
- MATH[x] selects a math waveform source.
- REF[x] selects a reference waveform source.

{ M[n]{A|B} | MATH[x] | REF[x] } is the optional secondary source for the measurement (e.g., for delay measurements). It follows the conventions of the primary source.

{ MEAS[x] } is the optional measurement ID the user wants to create the measurement on.

See “:MEASUrement:MEAS<x>:SOUrce<x>” for details on valid source names

Examples
MEASUREMENT:ADDMeas "PAM4", “RLM”,REF1 adds a RLM measurement on Ref1
MEASUREMENT:ADDMeas "PAM4", “RLM”,M1A, MEAS7 adds a RLM measurement on M1A with measurement ID 7
MEASUREMENT:ADDMeas "PAM4", “RLM”,M1A, M1B, MEAS10 adds a RLM measurement on M1A with measurement ID 10
MEASUREMENT:ADDMeas “PAM4”, “TDECQ”, M1A, MEAS1 adds a TDECQ measurement on M1A with measurement ID 1
MEASUREMENT:MEAS1:PLOT:STATe "Equalized Eye", ON; add an Equalized Eye diagram for the TDECQ measurement with measurement ID 1

Limitations
The selected source must exist and be active for the measurement to be created successfully.

:MEASUrement:MEAS<x>:TYPe?

Description
This query-only command returns a measurement type as a string, for a measurement specified by <x>.

Syntax
MEASUrement:MEAS<x>:TYPe?

Returns
Type of the given measurement

Examples
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>:SOUrce<y>

Description

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>. 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.

Syntax
MEASUrement:MEAS<x>:SOUrce<y> <source>

Arguments
<source> can be one of:
M<n>A|B selects a channel waveform as the source or destination waveform.
MATH<x> selects a math waveform as the source
REF<x> selects a reference waveform as the source

Returns
the quoted string source of the specified measurement

Examples
MEASUREMENT:MEAS2:SOURCE1 MATH1 sets MATH1 as the source waveform for Measurement 2
MEASUREMENT:MEAS7:SOURCE1 M2A sets channel A on module 2 as the source waveform for Measurement 7
MEASUREMENT:MEAS7:SOURCE1? might return MEASUREMENT:MEAS7:SOURCE1 REF1 indicating that the first source for Measurement 7 is Ref 1

:MEASUrement:MEAS<x>:LABel

Description
This command sets or queries the label for the measurement. The measurement number is specified by <x>.

Syntax
MEASUrement:MEAS<x>:LABel <QString>

Arguments
<QString> is the quoted string measurement label.

Returns
the quoted string label of the specified measurement

Examples
MEASUREMENT:MEAS1:LABel "Delay" sets the label to Delay.
MEASUrement:MEAS1:LABel? might return :MEASUREMENT:MEAS1:LABEL "Peak-to-Peak" indicating that the measurement 1 label is Peak-to-peak.

:MEASUrement:MEAS<x>:VALue? [<string_attribute>] (Query Only)

Description
This query-only command returns the value that is calculated for the measurement specified by <x>.

Syntax
MEASUrement:MEAS<x>:VALue? [<string_attribute>]

Arguments
[<string_attribute>] optionally the quoted string name of the desired result attribute. This is required for measurements with multiple attributes. Measurements with a single result do not require an attribute to be specified.

Returns
NR3 the value of the specified measurement for present acquisition if source is reference waveform
NR3 the value of the specified measurement across acquisitions if source is live waveform

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:MEAS4:VALUE? "L3" might return MEASUREMENT:MEAS4:VALUE "L3",5.89248655395E-003, indicating that the value for the "L3" attribute of Meas 4 is 5.892 mV

:MEASUrement:MEAS<x>:MAXimum? [<string_attribute>] (Query Only)

Description
This query only command returns the maximum value found for the measurement slot specified by x, since the last statistical reset. Measurements with a single result do not require an attribute to be specified. Measurements with multiple attributes require a specified attribute.
Tip: To find available result attributes for a measurement, use the query MEASUrement:MEAS[x]:RESult:ATTR?

Syntax
MEASUrement:MEAS<x>:MAXimum? [<string_attribute>]

Arguments
[<string_attribute>] optionally the quoted string name of the desired result attribute. This is required for measurements with multiple attributes. Measurements with a single result do not require an attribute to be specified.

Returns
NR3 the maximum value for present acquisition if source of measurement is reference waveform.
NR3 the maximum value across acquisitions if source of measurement is live waveform.

Examples

MEASUREMENT:MEAS3:MAXIMUM? might return MEASUREMENT:MEAS3:MAXIMUM 4.27246105395E-003, indicating that the maximum value for Meas 3 is 4.272 mV
MEASUREMENT:MEAS9:MAXIMUM? “L3” might return MEASUREMENT:MEAS9:MAXIMUM “L3”,7.23248678995E-003, indicating that the maximum value for the “L3” attribute of Meas 9 is 7.232 mV

:MEASUrement:MEAS<x>:GATing:STATE

Description
This command sets or queries the gating setting for the given measurement. Measurements are specified by <x>. This command is equivalent to opening the Measurement configuration menu and setting gating to enabled ON of OFF.

Syntax
MEASUrement:MEAS<x>:GATing:STATE { ON | OFF | 0 | 1 }
MEASUrement:MEAS<x>:GATing:STATE?

Arguments
ON or any other non-zero value enables gating
OFF or 0 disables gating

Returns
0 or 1 indicating state of gating of the specified measurement is OFF or ON respectively

Examples
MEASUREMENT:MEAS2:GATING:STATE ON sets gating for MEAS2’s gating to enabled (ON)
MEASUREMENT:MEAS1:GATING:STATE OFF sets gating for MEAS1’s gating to enabled (OFF)
MEASUREMENT:MEAS3:GATING:STATE 0 sets gating for MEAS3’s gating to enabled (OFF)
MEASUREMENT:MEAS2:GATING:STATE? Might return MEASUREMENT:MEAS2:GATING:STATE 0 meaning that gating for MEAS2’s is disabled (OFF)

:MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)

Description
This command returns a list of measurement specific attributes by name for the given measurement. Measurements are specified by <x>. This command is equivalent to double-clicking the measurement badge, opening the Measurement configuration subcategory in the menu, and viewing configuration attributes.

Syntax
MEASUrement:MEAS<x>:CONfig:ATTRibutes?

Returns
a comma separated list of configuration attribute names or an empty string if the specified measurement has no specified attributes

Examples
:MEASUREMENT:MEAS1:CONFIG:ATTRIBUTES? might
return :MEASUREMENT:MEAS1:CONFIG:ATTRIBUTES "TrackingMethod" to indicate the measurement configuration attribute is "TrackingMethod".

:MEASUrement:MEAS<x>:CONfig <string_attribute>,<value>

Description

This command returns or sets the value of a measurement specific configuration attribute for the given measurement. Measurements are specified by <x>. The configuration attribute is specified by its string name. The query returns the attribute string name and the value of a configuration attribute separated by a comma. The command sets the configuration value to the input value, if the input is valid (of correct type and/or range, where applicable), and the specific attribute is configurable (in some cases attributes may be read-only). This command is equivalent to double-clicking the measurement badge, opening the Measurement configuration subcategory in the menu, viewing configuration attributes and their values, and setting a non-read-only configuration attribute.
Tip: Measurement attributes are specific to each measurement, use the query MEASUrement:MEAS<x>:CONfig:ATTRibutes? to find a measurement's available attributes.

Syntax
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
MEASUrement:MEAS<x>:CONfig? <string_attribute>
Arguments
<string_attribute> is the measurement attribute name as a quoted string
<QString> if applicable, an attribute may be set to a string value
<NR3> if applicable, an attribute may be set to a numeric value
Returns
The value of the configuration attribute as an NR3, quoted string, or boolean.

Examples
:MEASUrement:MEAS1:CONfig "TrackingMethod","Min/Max" sets the Tracking Method to Min/Max
:MEASUrement:MEAS1:CONfig? "TrackingMethod" might return :MEASUREMENT:MEAS1:CONFIG "TrackingMethod","Auto"

MEASUrement:MEAS<x>:RESults:ATTRibutes? (Query only)

Description
This command returns a list of measurement specific attributes by name for the given measurement. Measurements are specified by <x>. This command is equivalent to double-clicking the measurement badge, opening the Measurement configuration subcategory in the menu, and viewing configuration attributes.

Syntax
MEASUrement:MEAS<x>:RESults:ATTRibutes?

Returns
A string containing a comma separated list of result attributes or an empty string if no result attributes are specified for the measurement.

Examples
MEASUREMENT:MEAS2:RESULTS:ATTRIBUTES? might return
MEASUREMENT:MEAS2:RESULTS:ATTRIBUTES "Level1,Level2,Level3,Level4"
MEASUREMENT:MEAS2:RESULTS:ATTRIBUTES? might return
MEASUREMENT:MEAS2:RESULTS:ATTRIBUTES "" indicating no result attributes are specified

MEASUrement:MEAS<x>:GATE[1|2]:PCTPOS

Description
This command or query sets or returns a measurement's gate's position. Chosen Measurement is specified by value given for <x>. The Gate can be 1 or 2.
Tip: a gate cannot be set unless gating for that measurement has been enabled
(with MEASUrement:MEAS<x>:GATing ON command)

Syntax
MEASUrement:MEAS<x>:GATE[1|2]:PCTPOS?
MEASUrement:MEAS<x>:GATE[1|2]:PCTPOS <NR1>

Arguments
<NR1> a numeric value for the position of the specified gate as a percentage

Returns
A numeric value for the position of the specified gate as a percentage

Examples
MEASUREMENT:MEAS2:GATE1:PCTPOS 27 sets gate position to 27% of total display
MEASUREMENT:MEAS2:GATE1:PCTPOS? Could return MEASUREMENT:MEAS1:GATE1:PCTPOS 27 if position was set to 27% of total display

:MEASUrement:MEAS<x>:MINimum? [<string_attribute>] (Query Only)

Description
This query only command returns the minimum value found for the measurement slot specified by x, since the last statistical reset. Measurements with a single result do not require an attribute to be specified. Measurements with multiple attributes require a specified attribute.
Tip: To find available result attributes for a measurement, use the query MEASUrement:MEAS[x]:RESult:ATTR?

Syntax
MEASUrement:MEAS<x>:MINimum? [<string_attribute>]

Arguments
[<string_attribute>] optionally the quoted string name of the desired result attribute. This is required for measurements with multiple attributes. Measurements with a single result do not require an attribute to be specified.

Returns
NR3 the minimum value for present acquisition if source of measurement is reference waveform.
NR3 the minimum value across acquisitions if source of measurement is live waveform.

Examples
MEASUREMENT:MEAS4:MINIMUM? might return MEASUREMENT:MEAS4: MINIMUM 4.27246105395E-003, indicating that the minimum value for Meas 4 is 4.272 mV
MEASUREMENT:MEAS3:MINIMUM? "L3" might return MEASUREMENT:MEAS3:MINIMUM "L3",5.89248655395E-003, indicating that the minimum value for the "L3" attribute of Meas 3 is 5.892 mV

:MEASUrement:MEAS<x>:MEAN? [<string_attribute>] (Query Only)

Description
This query only command returns the mean value found for the measurement slot specified by x, since the last statistical reset. Measurements with a single result do not require an attribute to be specified. Measurements with multiple attributes require a specified attribute.
Tip: To find available result attributes for a measurement, use the query MEASUrement:MEAS<x>:RESult:ATTR?

Syntax
MEASUrement:MEAS<x>:MEAN? [<string_attribute>]

Arguments
[<string_attribute>] optionally the quoted string name of the desired result attribute. This is required for measurements with multiple attributes. Measurements with a single result do not require an attribute to be specified.

Returns
NR3 the mean value for present acquisition if source of measurement is reference waveform
NR3 the mean value across acquisitions if source of measurement is live waveform

Examples
MEASUREMENT:MEAS2:MEAN? might return MEASUREMENT:MEAS2:MEAN 3.14146105395E-003, indicating that the mean value for Meas 2 is 3.141 mV
MEASUREMENT:MEAS4:MEAN? “L3” might return MEASUREMENT:MEAS4:MEAN “L3”,4.12348655395E-003, indicating that the mean value for the “L3” attribute of Meas 4 is 4.123 mV

:MEASUrement:MEAS<x>:STDdev? [<string_attribute>] (Query Only)

Description
This query only command returns the standard deviation value found for the measurement slot specified by x, since the last statistical reset. Measurements with a single result do not require an attribute to be specified. Measurements with multiple attributes require a specified attribute.
Tip: To find available result attributes for a measurement, use the query MEASUrement:MEAS<x>:RESult:ATTR?

Syntax
MEASUrement:MEAS<x>:STDdev? [<string_attribute>]

Arguments
[<string_attribute>] optionally the quoted string name of the desired result attribute. This is required for measurements with multiple attributes. Measurements with a single result do not require an attribute to be specified.

Returns
NR3 the standard deviation value for present acquisition if source of measurement is reference waveform.
NR3 the standard deviation value across acquisitions if source of measurement is live waveform.

Examples
MEASUREMENT:MEAS2:STDdev? might return MEASUREMENT:MEAS2:STDDEV 5.80230767128E–009, indicating that the standard deviation value for Meas 2 is 5.80 ns.
MEASUREMENT:MEAS4:STDdev? “L3” might return MEASUREMENT:MEAS4:STDDEV “L3”,1.16796169259E-011, indicating that the standard deviation for the “L3” attribute of Meas 4 is 11.68 ps.

:MEASUrement:MEAS<x>:PK2PK? [<string_attribute>] (Query Only)

Description
This query only command returns the peak-to-peak value found for the measurement slot specified by x, since the last statistical reset. Measurements with a single result do not require an attribute to be specified. Measurements with multiple attributes require a specified attribute.
Tip: To find available result attributes for a measurement, use the query MEASUrement:MEAS<x>:RESult:ATTR?

Syntax

MEASUrement:MEAS<x>:PK2PK? [<string_attribute>]

Arguments
[<string_attribute>] optionally the quoted string name of the desired result attribute. This is required for measurements with multiple attributes. Measurements with a single result do not require an attribute to be specified.

Returns
NR3 the peak-to-peak value for present acquisition if source of measurement is reference waveform.
NR3 the peak-to-peak value across acquisitions if source of measurement is live waveform.

Examples
MEASUREMENT:MEAS2:PK2PK? might return MEASUREMENT:MEAS2:PK2PK 200.0E-3 indicating the peak-to-peak value for Meas 2 is 200 mV.
MEASUREMENT:MEAS4:PK2PK? “L3” might return MEASUREMENT:MEAS4:PK2PK “L3”,4.000E-3, indicating that the peak-to-peak value for the “L3” attribute of Meas 4 is 400 mV.

:MEASUrement:MEAS<x>:DELEte

Description
This command deletes the specified measurement. If the measurement specified by <x> does not exist or is not able to be deleted, an error will be reported.

Syntax
MEASUrement:MEAS<x>:DELEte

Examples
MEASUREMENT:MEAS2:DELETE will delete measurement 2.

:MEASUrement:DELEte:ALL

Description
This command deletes all measurements. If a measurement is not able to be deleted, an error will be reported.

Syntax
MEASUrement:DELEte:ALL

Examples
MEASUREMENT:DELETE:ALL will delete all measurements.

:MEASUrement:MEAS<x>:COUNt? [<string_attribute>] (Query Only)

Description
This query only command returns the count of the result values found for the measurement slot specified by x, since the last statistical reset. Measurements with a single result do not require an attribute to be specified. Measurements with multiple attributes require a specified attribute.
Tip: To find available result attributes for a measurement, use the query MEASUrement:MEAS<x>:RESult:ATTR?

Syntax
MEASUrement:MEAS<x>:COUNt? [<string_attribute>]

Arguments

[<string_attribute>] optionally the quoted string name of the desired result attribute. This is required for measurements with multiple attributes. Measurements with a single result do not require an attribute to be specified.

Returns
An integer value count of the result values of a given measurement or measurement's attribute for present acquisition if source is reference waveform.
An integer value count of the result values of a given measurement or measurement's attribute across acquisitions if source is live waveform.

Examples
MEASUREMENT:MEAS3:COUNT? might return MEASUREMENT:MEAS3:COUNT 1, indicating that the count of result values for Meas 3 is 1
MEASUREMENT:MEAS9:COUNT? “L3” might return MEASUREMENT:MEAS9:COUNT “L3”,39, indicating that the count of result values for the “L3” attribute of Meas 9 is 39

:MEASUrement:MEAS<x>:STATus? (Query Only)

Description
This query-only command returns a measurement Info (Error/Warning) as a string, for a measurement specified by <x>.

Syntax
:MEASUrement:MEAS<x>:STATus?

Returns
Error/Warning information of the given measurement.

Examples
MEASUREMENT:MEAS1:STATus? might return Error/Warning for the selected measurement, if any.

:MEASUrement:MEAS<n>:PLOT:STATe

Description
This command sets or gets the state of the plot named <plot_name> for measurement specified by <n>.

Syntax
MEASUrement:MEAS<n>:PLOT:STATe <plot_name>,{ ON | OFF | 0 | 1 }
MEASUrement:MEAS<n>:PLOT:STATe? <plot_name>

Arguments
<plot_name> is the plot attribute name as a quoted string
ON or any other non-zero value enables the plot
OFF or 0 disables the plot

Returns
0 or 1 indicating state of the plot of the specified measurement is OFF or ON respectively

Examples
MEASUREMENT:MEAS2:PLOT:STATE "Equalized Eye",ON turns on the Equalized Eye plot when TDECQ measurement is added as MEAS2
MEASUREMENT:MEAS1:PLOT:STATE "Equalized Eye",0 turns off the Equalized Eye plot when TDECQ measurement is added as MEAS1

MEASUREMENT:MEAS1:PLOT:STATE? "Equalized Eye" might
return MEASUREMENT:MEAS1:PLOT:STATE "Equalized Eye",0 meaning that the plot for MEAS1 is turned off

MEASUrement:MEAS<x>:RLEVel:ATTRibutes? (Query Only)

Description
This query only command returns a list of the names of the ref level attributes available for the measurement slot specified by x.

Syntax
MEASUrement:MEAS<x>:RLEVel:ATTRibutes?

Related Commands
MEASUrement:MEAS<x>:RLEVel [<stringAttributes>]

Returns
A comma separated list of the names of the ref level attributes available for the measurement slot specified by x

Examples
MEASUREMENT:MEAS3:RLEVEL:ATTRIBUTES? might return
MEASUREMENT:MEAS3:RLEVEL:ATTRIBUTES "High","Mid","Low", indicating that there are 3 reference levels, named "High","Mid", and "Low", available to query and set for the measurement 3
MEASUREMENT:MEAS9:RLEVEL:ATTRIBUTES? might return
MEASUREMENT:MEAS9:RLEVEL:ATTRIBUTES "Mid", indicating that there is a single reference level with the name "Mid" available to query and set for Measurement 9.

MEASUrement:MEAS<x>:RLEVel:METHod

Description
This command sets or queries the method the instrument uses to calculate the reference levels for a specified measurement taken on a specified source waveform. The measurement slot is specified by x.

Syntax
MEASUrement:MEAS<x>:RLEVel:METHod {RELative | ABSolute }
MEASUrement:MEAS<x>:RLEVel:METHod?

Related Commands
MEASUrement:MEAS<x>:RLEVel?

Arguments
- **RELative** calculates the reference levels as a percentage of the High/Low amplitude (High amplitude minus the Low amplitude). The default values are 90% for the high reference level, 10% for the low reference level, and 50% for the mid reference levels. You can set other percentages using the MEASUrement:MEAS:RLEVel:RELative commands.
- **ABSolute** uses reference levels set explicitly in absolute user units with the MEASUrement:MEAS:RLEVel:ABSolute commands (see related commands above). This method is useful when precise values are required (for example, when you are designing to published interface specifications such as RS-232-C). The default values are 0 V for the high reference level, the low reference level, and the mid reference levels.

Returns
RELATIVE or ABSOLUTE

Examples
MEASUREMENT:MEAS1:RLEVEL:METHOD RELATIVE sets the method of calculating the reference levels to relative for Measurement 1;
MEASUREMENT:MEAS8:RLEVEL:METHOD? might return MEASUREMENT:MEAS8:RLEVEL:METHOD ABSOLUTE, indicating the reference levels used are set to absolute values in user units.

MEASUrement:MEAS<x>:RLEVel <stringAttribute>

Description
This command sets or queries reference level for the specified measurement.
If the reference level method is set to ABSOLUTE, this command will set or query the given reference level in absolute user units for the specified measurement.
If the reference level method is set to RELATIVE, this command will set or query the value as a percent of the High/Low range that the instrument uses to calculate the given reference level for the specified measurement, where 100% is equal to the High/Low range.
The measurement slot is specified by x. And the reference level is specified by it's <stringAttribute> name.
When a measurement has multiple reference levels the reference level attribute name must be specified. If the measurement has a single reference level, providing the attribute name is not required.

Tip: To Find a list of available reference level attribute names for the given measurement use the MEASUrement:MEAS<x>:RLEVel:ATTRibutes? query.
Set or query the reference level method by using the command MEASUrement:MEAS<x>:RLEVel:METHod

Syntax
MEASUrement:MEAS<x>:RLEVel [<stringAttribute>],<NR3>
MEASUrement:MEAS<x>:RLEVel [<stringAttribute>]

Related Commands
MEASUrement:MEAS<x>:RLEVel:METHod
MEASUrement:MEAS<x>:RLEVel:ATTRibutes?

Arguments
<stringAttribute> is the reference level by name to set or query
NR3 can be from 0 to 100 (percent) and is the given reference level.

Returns
When Method is set to ABSOLUTE, NR3 is the given reference level in absolute user units.
When in Method is set to RELATIVE, NR3 is the given reference level as a percentage (value 0-100) of the High/Low Range.

Examples
When the Reference Level method is set to RELATIVE the command MEASUREMENT:MEAS3:RLEVEL "High",20 sets the "High" reference level for Measurement 3 to 20% of the High/Low range.
When the Reference Level method is set to RELATIVE the query MEASUREMENT:MEAS2:RLEVEL? "Mid" might return MEASUREMENT:MEAS2:RLEVEL "Mid",10, indicating that the "Mid" reference level for Measurement 2 is set to 10% of the High/Low range.

When the Reference Level method is set to ABSOLUTE the command MEASUREMENT:MEAS3:REFLEVEL "High",4.0E–2 sets the "High" reference level for Measurement 3 to 40 mV.

When the Reference Level method is set to ABSOLUTE the query MEASUREMENT:MEAS2:REFLEVEL? "Mid" might return MEASUREMENT:MEAS2:REFLEVEL "Mid",5.0000000000E–2, indicating that the "Mid" reference level for Measurement 2 is set to 50 mV.

MEASUrement:MEAS<x>:CONfig:ATTRibutes?

Description
This command returns the value of a measurement specific configuration attribute for the given measurement. Measurements are specified by <x>. The query returns the attribute string name and the value of a configuration attribute separated by a comma.

Syntax
MEASUrement:MEAS<x>:CONfig:ATTRibutes?
Returns
The value of the configuration attribute as an NR3, quoted string, or boolean.

Examples
:MEASUrement:MEAS1:CONfig:ATTRibutes? might return "TrackingMethod","Auto".

:MEASUrement:MEAS<x>:CONfig <string_attribute>,<value>

Description

This command returns or sets the value of a measurement specific configuration attribute for the given measurement. Measurements are specified by <x>. The configuration attribute is specified by its string name.
Measurement specific attribute(s): "TrackingMethod"
Measurement specific attribute values: "Auto", "Mean", "Mode", "Min/Max"

Syntax
:MEASUrement:MEAS<x>:CONfig <string_attribute>,<value>
Returns
The value of the configuration attribute as quoted string.

Examples
:MEASUrement:MEAS1:CONfig:ATTRibutes? might return "TrackingMethod","Auto".

:MEASUrement:ADDMEAS "PULSE","PCross",<source1>[,<source2>]

See :MEASUrement:ADDMEAS

:MEASUrement:ADDMEAS "PULSE","PWidth",<source1>[,<source2>]

See :MEASUrement:ADDMEAS

:MEASUrement:ADDMEAS "PULSE","RMSJitter",<source1>[,<source2>]

See :MEASUrement:ADDMEAS

:MEASUrement:ADDMEAS "PULSE","Pk-PkJitter",<source1>[,<source2>]

See :MEASUrement:ADDMEAS

:MEASUrement:ADDMEAS "PULSE","Delay",<source1>,<source2>

See :MEASUrement:ADDMEAS

:MEASUrement:ADDMEAS "PULSE","NCross",<source1>[,<source2>]
See :MEASUrement:ADDMEAS

Adding Level deviation measurement on a source

Description

This command adds level deviation measurement from PAM4 category on the given source with specified measurement ID

Syntax
MEASUREMENT:ADDMeas "PAM4","LDeviation",{M[n]{A|B} | REF[x]}[, MEAS[x]]

Examples
MEASUREMENT:ADDMeas "PAM4","LDeviation",M1A
MEASUREMENT:ADDMeas "PAM4","LDeviation",Ref1
MEASUREMENT:ADDMeas "PAM4","LDeviation",Ref1,MEAS20

Adding Level Thickness measurement on a source

Description

This command adds level thickness measurement from PAM4 category on the given source with specified measurement ID

Syntax
MEASUREMENT:ADDMeas "PAM4","LThickness",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "PAM4","LThickness",M1A
MEASUREMENT:ADDMeas "PAM4","LThickness",Ref1
MEASUREMENT:ADDMeas "PAM4","LThickness",Ref1,MEAS2

Adding Eye Width measurement on a source

Description

This command adds Eye width measurement from PAM4 category on the given source with specified measurement ID

Syntax
MEASUREMENT:ADDMeas "PAM4","EyeWidth",{M[n]{A|B} | REF[x]}[, MEAS[x]]

Examples
MEASUREMENT:ADDMeas "PAM4","EyeWidth",M1A
MEASUREMENT:ADDMeas "PAM4","EyeWidth",Ref1
MEASUREMENT:ADDMeas "PAM4","EyeWidth",Ref1,MEAS20

Querying results of Eye Width measurement

Description

Eye width measurement gives following results for all 3 PAM4 eyes
- Threshold

- Width

Syntax
MEASUREMENT:MEAS<x>VALue? [<string_attribute>] (Query Only)

Arguments
String attribute will be
 Upper Eye threshold -- "ThreshU"
 Upper Eye width - "WidthU"
 Middle Eye threshold - "ThreshM"
 Middle Eye width - "WidthM"
 Lower Eye threshold - "ThreshL"
 Lower Eye width - "WidthL"

Returns
"ThreshU" returns the threshold of upper eye at which eye width is computed
"WidthU" returns the width of upper eye at the "ThreshU"
"ThreshM" returns the threshold of upper eye at which eye width is computed
"WidthM" returns the width of upper eye at the "ThreshM"
"ThreshL" returns the threshold of upper eye at which eye width is computed
"WidthL" returns the width of upper eye at the "ThreshL"

Examples
MEASUREMENT:MEAS1:VALUE? "ThreshU"
MEASUREMENT:MEAS1:VALUE? "WidthM"

Adding Eye Height measurement on a source

Description

This command adds Eye height measurement from PAM4 category on the given source with specified measurement ID

Syntax
MEASUREMENT:ADDMeas "PAM4","EyeHeight",{M[n]{A|B} | REF[x]}[, MEAS[x]]
Examples
MEASUREMENT:ADDMeas "PAM4","EyeHeight",M1A
MEASUREMENT:ADDMeas "PAM4","EyeHeight",Ref1
MEASUREMENT:ADDMeas "PAM4","EyeHeight",Ref1,MEAS20

Querying results of Eye height measurement

Description

Eye Height measurement gives following results for all 3 PAM4 eyes
- Offset
- Height

Syntax
MEASUREMENT:MEAS<x>VALue? [<string_attribute>] (Query Only)

Arguments
String attribute will be
 Upper Eye Offset -- "OffsetU"
 Upper Eye Height - "HeightU"
 Middle Eye Offset -- "OffsetM"

Middle Eye Height - "HeightM"
Lower Eye Offset -- "OffsetM"
Lower Eye Height - "HeightM"

Returns
"OffsetU" returns the offset of upper eye at which eye height is computed
"HeightU" returns the height of upper eye at the "OffsetU"
"OffsetM" returns the offset of upper eye at which eye height is computed
"HeightM" returns the height of upper eye at the "OffsetM"
"OffsetL" returns the offset of upper eye at which eye height is computed
"HeightL" returns the height of upper eye at the "OffsetL"

Examples
MEASUREMENT:MEAS1:VALUE? "OffsetU"
MEASUREMENT:MEAS1:VALUE? "HeightL"

Adding PAM4 Summary on a source

Description

This command adds PAM4 Summary measurement from PAM4 category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "PAM4","PAM4Summary",{M[n]{A|B} | REF[x]}[, MEAS[x]]

Examples

MEASUREMENT:ADDMeas "PAM4","PAM4Summary",M1A
MEASUREMENT:ADDMeas "PAM4","PAM4Summary",Ref1
MEASUREMENT:ADDMeas "PAM4","PAM4Summary",Ref1,MEAS20

For Other supported commands(below given), please refer general measurement command group.
- Setting label
- Setting source
- Querying results
- Querying statistics of present acquisition

Changing or querying measurement configurations

Description

PAM4 Summary has two configurations that can be set or queried.

- ERAdjustPct: Extinction Ratio adjust/correction factor in percentage. It is used by Extinction Ratio measurement in PAM4 Summary
- NLOutput: Normalized output. It is used by Levels measurement in PAM4 Summary
- RLMMethod: Method for computing of RLM
- RLMLevelsMethod: Method of computing Levels measurement for RLM
- AOPUnits: Units for AOP.

Syntax
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: "ERAdjustPct,NLOutput,RLMMethod,AOPUnits,RLMLevelsMethod"

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}

Arguments
ERAdjustPct can take any double value from -100 to 100
NLOutput can take boolean value(True or False)
RLMMethod can take the following values: "Cl. 120D.3.1.2", "Cl. 94.3.12.5.1", "All"
AOPUnits can take the following values: "W", "dBm"
RLMLevelsMethod can take the following values: "Central sample from each UI", "Central 2UI of longest run length"

Returns
Configuration query will return the value set before.
Querying "ERAdjustPct" configuration will return the double value set before.
Querying "NLOutput" configuration will return the boolean value set before.
Querying "RLMMethod" configuration will return the string value set before.
Querying "AOPUnits" configuration will return the string value set before.
Querying "RLMLevelsMethod" configuration will return the string value set before.

Examples

Setting Configurations:
MEASUrement:MEAS1:CONfig "ERAdjustPct",10.5
MEASUrement:MEAS1:CONfig "NLOutput",0
MEASUrement:MEAS1:CONfig "NLOutput",ON
MEASUrement:MEAS1:CONfig "NLOutput",OFF
MEASUrement:MEAS1:CONfig "RLMMethod","Cl. 120D.3.1.2"

Querying Configurations:
MEASUrement:MEAS1:CONfig? "ERAdjustPct"
MEASUrement:MEAS1:CONfig? "NLOutput"
MEASUrement:MEAS1:CONfig? "RLMMethod"

Querying results of PAM4 Summary measurement

Description
PAM4 Summary gives following results

- RLM(Cl. 120D.) - If RLM method is set to All or Cl. 120D.3.1.2
- RLM(Cl. 94.) - If RLM method is set to All or Cl. 94.3.12.5.1
- OMAouter
- ER
- AOP
- T Time
- L3
- L2
- L1
- L0

Syntax
Querying PAM4 Summary result attributes is done using below command
MEASUREMENT:MEAS<x>:RESULTS:ATTRIBUTES?

Querying result of an attribute of PAM4 Summary is done using
MEASUREMENT:MEAS<x>VALue? [<string_attribute>] (Query Only)

String attribute can be:

- RLM(Cl. 120D.) - If RLM method is set to All or Cl. 120D.3.1.2
- RLM(Cl. 94.) - If RLM method is set to All or Cl. 94.3.12.5.1
- OMAouter
- ER
- AOP
- T Time
- L3
- L2
- L1
- L0

Returns
PAM4 Summary result attributes query will return "RLM(Cl. 120D.),RLM(Cl. 94.),OMAouter,ER,AOP,T Time,L3,L2,L1,L0"

Depending on the Measurement configuration, RLM(Cl. 120D.) or RLM(Cl. 94.) may not be present.

Querying result of an attribute of PAM4 Summary and return value are given in below table.

string attribute	return value
RLM(Cl. 120D.)	returns the RLM(Cl. 120D.) result with no unit
RLM(Cl. 94.)	returns the RLM(Cl. 94.) result with no unit
OMAouter	returns the OMA outer result with unit as dBm
ER	returns the extinction ratio result with unit configured by user
AOP	returns the AOP result with unit as ordinate unit of input signal
T Time	returns the transition time result with unit as abscissa unit of input signal
L3	returns the mean of level3 with unit configured by user(normalized or absolute)
L2	returns the mean of level2 with unit configured by user(normalized or absolute)
L1	returns the mean of level1 with unit configured by user(normalized or absolute)
L0	returns the mean of level0 with unit configured by user(normalized or absolute)

Examples
```
MEASUREMENT:MEAS1:VALUE? "RLM(Cl. 120D.)"
MEASUREMENT:MEAS1:VALUE? "OMAouter"
MEASUREMENT:MEAS1:VALUE? "ER"
MEASUREMENT:MEAS1:VALUE? "AOP"
MEASUREMENT:MEAS1:VALUE? "T Time"
MEASUREMENT:MEAS1:VALUE? "L3"
MEASUREMENT:MEAS1:VALUE? "L2"
MEASUREMENT:MEAS1:VALUE? "L1"
MEASUREMENT:MEAS1:VALUE? "L0"
```

Adding TPE measurement

Description
This command adds TPE measurement from PAM4 category on the given source with specified measurement ID.

Syntax
MEASurement:ADDMeas "PAM4", "TPE", <source> <measX>

Changing or querying TPE measurement configurations

Use the following query to get to know the available configurations for the measurement.
MEASUREMENT:MEAS<x>:CONFIG:ATTRIBUTES? (Query only)

Returns
:MEASUREMENT:MEAS1:CONFIG:ATTRIBUTES "HitRatio,TPEUnits,TPEat"

Use the following syntax to set and query the configuration:

Hit ratio:
MEASUREMENT:MEAS1:CONFIG "HitRatio",1e-5 (Set)
MEASUREMENT:MEAS1:CONFIG? "HitRatio" (Query)

TPE unit:
MEASUREMENT:MEAS1:CONFIG "TPEUnits", "W" (Set)
MEASUREMENT:MEAS1:CONFIG? "TPEUnits" (Query)

TPE at:
MEASUREMENT:MEAS1:CONFIG "TPEat", "Level-0" (Set)
MEASUREMENT:MEAS1:CONFIG "TPEat", "Level-3" (Set)
MEASUREMENT:MEAS1:CONFIG? "TPEat " (Query)

Querying results of TPE measurement

Use the following query to get to know the available attribute for the measurement.
MEASUREMENT:MEAS<x>:RESULTS:ATTRIBUTES? (Query only)

Returns
:MEASUREMENT:MEAS1:RESULTS:ATTRIBUTES "Oversh.HR,Undersh.HR,TPE"

Use the following syntax to query the results:

Overshoot:
MEASUREMENT:MEAS<x>:VALUE? "Oversh.HR" (Query)

Result
:MEASUREMENT:MEAS1:VALUE "Oversh.HR",<result>

Undershoot
MEASUREMENT:MEAS<x>:VALUE? "Undersh.HR" (Query)

Result
:MEASUREMENT:MEAS1:VALUE "Undersh.HR",<result>

TPE
MEASUREMENT:MEAS<x>:VALUE? "TPE" (Query)

Result
:MEASUREMENT:MEAS1:VALUE "TPE",<result>

Adding TDECQ measurement on a source

Description
This command adds TDECQ measurement from PAM4 category on the given source with specified measurement ID.

Syntax
MEASurement:ADDMEAS “PAM4”,”TDECQ”,{M[n]{A|B} | REF[x]}[, MEAS[x]]

Changing or querying TDECQ measurement configurations

Use the following query to get to know the available configurations for the measurement.
MEASUREMENT:MEAS<x>:CONFIG:ATTRIBUTES? (Query only)

Returns
:MEASUREMENT:MEAS<x>:CONFIG:ATTRIBUTES "VerticalThresholdAdjust,VerticalAdjustLimit,TargetSER,CeqIndB,HistogramWidth,HistogramSpacing,FFEAutoset,ExtendedSearch,FFERecalc,FFELockMainCursor,FFEMainCursorPosition,FFETaps,FFETapsPerUI,FFEMaxPrecursors,FFETapVal”

Use the following syntax to set and query the configuration:

MEASUREMENT:MEAS<x>:CONFIG “<parameter_name>”,<value>
MEASUREMENT:MEAS<x>:CONFIG> “<parameter_name>”

Parameter_name	**Type**	**Value**
VerticalThresholdAdjust	Boolean	1/true/ON or 0/false/OFF
VerticalAdjustLimit	Double	0 to 3
TargetSER	Double	1e-15 to 1e-2
CeqIndB	Boolean	1/true/ON or 0/false/OFF
HistogramWidth	Double	0.01 to 0.08
HistogramSpacing	Double	0.08 to 0.12
FFEAutoset	Boolean	1/true/ON or 0/false/OFF
ExtendedSearch	Boolean	1/true/ON or 0/false/OFF
FFERecalc	Boolean	1/true/ON or 0/false/OFF
FFELockMainCursor	Boolean	1/true/ON or 0/false/OFF
FFEMainCursorPosition	Integer	0 to FFE number of taps – 1
FFETaps	Integer	1 to 99
FFETapsPerUI	Integer	1 or 2
FFEMaxPrecursors	Integer	0 to FFE number of taps – 1
FFETapValues	Double array	It can take “FFETaps” number of values

If FFE is executed in MATH and MATH output is given as source for TDECQ, then configurations affecting only FFE are ignored while evaluating TDECQ.

Querying results of TDECQ measurement

Use the following query to get to know the available attribute for the measurement.
MEASUREMENT:MEAS<x>:RESULTS:ATTRIBUTES? (Query only)

Returns
:MEASUREMENT:MEAS<x>:RESULTS:ATTRIBUTES "AOP,TDECQ,Ceq,SERUpL,SERUpR,SERMidL,SERMidR,SERLoL,SERLoR"

Use the following syntax to query the results:

AOP:
MEASUREMENT:MEAS<x>:VALUE? "AOP" (Query only)

TDECQ:
MEASUREMENT:MEAS<x>:VALUE? "TDECQ" (Query only)

Ceq::
MEASUREMENT:MEAS<x>:VALUE? "Ceq" (Query only)

SER upper eye left
MEASUREMENT:MEAS<x>:VALUE? " SERUpL" (Query only)

SER upper eye right
MEASUREMENT:MEAS<x>:VALUE? " SERUpR" (Query only)

SER middle eye left
MEASUREMENT:MEAS<x>:VALUE? " SERMidL" (Query only)

SER middle eye right
MEASUREMENT:MEAS<x>:VALUE? " SERMidR" (Query only)

SER lower eye left
MEASUREMENT:MEAS<x>:VALUE? " SERLoL" (Query only)

SER lower eye right
MEASUREMENT:MEAS<x>:VALUE? " SERLoR" (Query only)

Adding NRZ Low measurement

Description
This command adds NRZ-Low measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","Low",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","Low",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","Low",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","Low",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: "TrackingMethod", "EyeAperture"

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}

Example:
MEASUrement:MEAS1:CONfig " EyeAperture”,10
MEASUrement:MEAS1:CONfig " TrackingMethod ",“Mean”

Adding NRZ High measurement

Description
This command adds NRZ-High measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","HIGH",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","HIGH",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","HIGH ",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","HIGH ",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: "TrackingMethod", "EyeAperture"

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig "EyeAperture",10
MEASUrement:MEAS1:CONfig "TrackingMethod",“Mean”

Adding NRZ ER measurement

Description
This command adds NRZ Extinction Ratio (ER) measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","Extinction Ratio",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","Extinction Ratio",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","Extinction Ratio",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","Extinction Ratio",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: "ERAdjust", "EyeAperture", “Units”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig "EyeAperture ",10
MEASUrement:MEAS1:CONfig "ERAdjust",“1.56”
MEASUrement:MEAS1:CONfig "Units",“%”

Adding NRZ Crossing Percentage measurement

Description
This command adds NRZ Crossing Percentage measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","PCTCROss",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","PCTCROss",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","PCTCROss",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","PCTCROss",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: "EyeAperture"

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig "EyeAperture ",10

Adding NRZ Crossing Level measurement

Description
This command adds NRZ Crossing Level measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","LEVCROss",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","LEVCROss",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","LEVCROss",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","LEVCROss",Ref1,MEAS20

Adding NRZ Crossing Time measurement

Description
This command adds NRZ Crossing Time measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","TIMCROss",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","TIMCROss",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","TIMCROss",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","TIMCROss",Ref1,MEAS20

Adding NRZ OMA measurement

Description
This command adds NRZ OMA measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","OMA",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","OMA",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","OMA",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","OMA",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “Units”, “Signal Ordinate”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig "Units",“Signal Ordinate”

Adding NRZ AC RMS measurement

Description
This command adds NRZ AC RMS measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","ACRMS",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","ACRMS",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","ACRMS",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","ACRMS",Ref1,MEAS20

Adding NRZ RMS Noise measurement

Description
This command adds NRZ RMS Noise measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","RMSNoise",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","RMSNoise",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","RMSNoise",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","RMSNoise",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.

MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “NoiseAt”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “NoiseAt”,”High”

MEASUrement:MEAS1:CONfig “NoiseAt”,”Low”

Adding NRZ Amplitude measurement

Description
This command adds NRZ Amplitude measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","Amplitude",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","Amplitude",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","Amplitude",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","Amplitude",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “TrackingMethod”, ”EyeAperture”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “EyeAperture”,10

MEASUrement:MEAS1:CONfig “TrackingMethod”,”Mean”

Adding NRZ Eye Width measurement

Description
This command adds NRZ Eye Width measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","EYEWIdth",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","EYEWIdth",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","EYEWIdth",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","EYEWIdth",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations

Return: ”EyeAperture”, ”Format”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “EyeAperature”,10

MEASUrement:MEAS1:CONfig “Format”,”Relative”

Adding NRZ RMS Jitter measurement

Description
This command adds NRZ RMS Jitter measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","RMSJitter",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","RMSJitter",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","RMSJitter",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","RMSJitter",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “TrackingMethod”, ”EyeAperture”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “EyeAperature”,10

MEASUrement:MEAS1:CONfig “TrackingMethod”,”Mean”

Adding NRZ Eye Height measurement

Description
This command adds NRZ Eye Height measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","EYEHeight",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","EYEHeight",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","EYEHeight",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","EYEHeight",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: ”EyeAperture”, ”Format”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “EyeAperature”,10

MEASUrement:MEAS1:CONfig “Format”,”Relative”

Adding NRZ Eye Pk-Pk Jitter measurement

Description
This command adds NRZ Eye Pk-Pk Jitter measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","PKPKJitter",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","PKPKJitter",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","PKPKJitter",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","PKPKJitter",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “TrackingMethod”, ”EyeAperture”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “EyeAperature”,10

MEASUrement:MEAS1:CONfig “TrackingMethod”,”Mean”

Adding NRZ Rise Time measurement

Description
This command adds NRZ Rise Time measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","RISe",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","RISe",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","RISe",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","RISe",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “TrackingMethod”, “EyeAperture”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}

Example:
MEASUrement:MEAS1:CONfig “TrackingMethod”,”Mode”

MEASUrement:MEAS1:CONfig “EyeAperature”,10

Adding NRZ Vertical Eye Closure (VECP) measurement

Description
This command adds NRZ VECP measurement from NRZ-Eye category on the given source with specified measurement ID

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","VECPenalty",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","VECPenalty",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","VECPenalty",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","VECPenalty",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: ”EyeAperture", “OMA Definition”, “User defined OMA”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig "EyeAperture", 50
MEASUrement:MEAS1:CONfig "UserdefinedOMAvalue”, 0.1W (range 1e-7W to 0.1W)

Adding NRZ Fall Time measurement

Description
This command adds NRZ Fall measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","FALL",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","FALL",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","FALL",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","FALL",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “TrackingMethod,EyeAperture”

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “TrackingMethod”,”Mode”

MEASUrement:MEAS1:CONfig “EyeAperature”,10

Adding NRZ TDEC measurement

Description
This command adds NRZ TDEC measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","TDEc",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","TDEc",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","TDEc",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","TDEc",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “TargetSER,HistogramWidth,HistogramSpacing,OMAdefinition,UserDefinedOMAvalue"

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “TargetSER”, 5e-5

MEASUrement:MEAS1:CONfig “HistogramWidth”,0.08

MEASUrement:MEAS1:CONfig “HistogramSpacing”,0.1

Adding NRZ Signal-to-Noise Ratio (SNR) measurement

Description
This command adds NRZ SNR measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","SNRatio",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","SNRatio",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","SNRatio",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","SNRatio",Ref1,MEAS20

Measurement specific configurations:
Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)
- This returns the list of available configurations
Return: “TrackingMethod,EyeAperature,NoiseAt"

Use the following syntax to set the configuration:
MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}
Example:
MEASUrement:MEAS1:CONfig “TrackingMethod,EyeAperature,NoiseAt”, “High”

MEASUrement:MEAS1:CONfig “TrackingMethod,EyeAperature,NoiseAt”, “Low”

MEASUrement:MEAS1:CONfig “TrackingMethod,EyeAperature,NoiseAt”, “High+Low”

Adding NRZ RMS measurement

Description
This command adds NRZ RMS measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","RMS",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","RMS",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","RMS",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","RMS",Ref1,MEAS20

Adding NRZ Bit Time measurement

Description
This command adds NRZ Bit Time measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","BITTime",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","BITTime",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","BITTime",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","BITTime",Ref1,MEAS20

Adding NRZ Bit Rate measurement

Description
This command adds NRZ Bit Rate measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","BITRate",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","BITRate",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","BITRate",Ref1
MEASUREMENT:ADDMeas "NRZ-EYE","BITRate",Ref1,MEAS20

Adding NRZ Duty Cycle Distortion measurement

Description
This command adds NRZ DCD measurement from NRZ-Eye category on the given source with specified measurement ID.

Syntax
MEASUREMENT:ADDMeas "NRZ-EYE","DISTDUty",{M[n]{A|B} | REF[x]}

Examples
MEASUREMENT:ADDMeas "NRZ-EYE","DISTDUty",M1A
MEASUREMENT:ADDMeas "NRZ-EYE","DISTDUty",Ref1

MEASUREMENT:ADDMeas "NRZ-EYE","DISTDUty",Ref1,MEAS20

Measurement specific configurations

Use the following query to get to know the available configurations for the measurement.
MEASUrement:MEAS<x>:CONfig:ATTRibutes? (Query only)

This returns the list of available configurations: "TrackingMethod", "EyeAperture", "Format"

Use the following syntax to set the configuration:

MEASUrement:MEAS<x>:CONfig <string_attribute>,{<QString>|<NR3>|<Boolean>}

Examples

MEASUrement:MEAS1:CONfig "TrackingMethod","Mean"

MEASUrement:MEAS1:CONfig "EyeAperture",50

MEASUrement:MEAS1:CONfig "Format",%"

For other supported commands (below given), please refer to the general measurement command group.
- Setting label
- Setting source
- Querying results
- Querying statistics of present acquisition

MEASUrement:MEAS<x>:CONFig:SNOise

Description
This command shall set or get the Scope Noise configuration parameter for the measurement.
This parameter shall be read-only if Measurement source is Live Module channel.
This parameter shall only be available for measurements where Scope Noise is used.

Syntax
MEASUrement:MEAS<x>:CONFig:SNOise <NR3>
MEASUrement:MEAS<x>:CONFig:SNOise?

Related Commands
MEASUrement:MEAS<x>:CONFig

Arguments
<NR3> is the numeric value of scope noise

Returns
The value of the scope noise associated with the measurement

Examples
MEASUrement:MEAS1:CONFig:SNOise 6e-6 shall set the scope noise associated with MEAS1 to 6 micro.
MEASUrement:MEAS1:CONFig:SNOise? may return 5e-6 to indicate 5 micro as the scope noise associated with MEAS1

Limitations
Only measurements which use scope noise as an input parameter shall show this value.
If the source of the measurement is a Live Module channel, then scope noise shall be read-only.

Mask Command Group

MASKS:ADD:USER

Description
This command adds a mask from the specified file on the given source with specified mask ID.

Syntax
:MASKS:ADD:USER <filename>,{ M[n]{A|B} | MATH[x] | REF[x] } [, MASK<x>]

Arguments
<filename> is the name of the file that contains the mask definition

{ M[n]{A|B} | MATH[x] | REF[x] } is the source for the mask test

- M[n]{A|B} selects a channel waveform source.
- MATH[x] selects a math waveform source.
- REF[x] selects a reference waveform source.

{ MASK<x>] } is the optional mask ID the user wants to create the mask on.

Returns
None

Examples
MASKS:ADD:USER "C:\10GTest.msk", M1A may add a mask defined in the file "C:\10GTest.msk" with mask ID MASK1 and M1A as source.
MASKS:ADD:USER "C:\10GTest.msk", M1A, MASK7 may add a mask defined in the file "C:\10GTest.msk" with mask ID MASK7 and M1A as source.

Limitations
The selected source must exist and be active for the mask to be created successfully.

To add Ethernet Masks:

MASKS:ADD:STAN <source>,"100GBASE_ER4","Ethernet"
MASKS:ADD:STAN <source>,"100GBASE_ER4FEC","Ethernet"
MASKS:ADD:STAN <source>,"100GBASE_LR4","Ethernet"
MASKS:ADD:STAN <source>,"100GBASE_LR4FEC","Ethernet"
MASKS:ADD:STAN <source>,"40GBASE_FR","Ethernet"
MASKS:ADD:STAN <source>,"40GBASE_LR4","Ethernet"
MASKS:ADD:STAN <source>,"40GBASE_SR4","Ethernet"
MASKS:ADD:STAN <source>,"100GBASE_SR10","Ethernet"

To add T11 Fibre Channel Masks:

MASKS:ADD:STAN <source>,"8GFC","T11 FibreChan"
MASKS:ADD:STAN <source>,"16GFCMMr6_1","T11 FibreChan"
MASKS:ADD:STAN <source>,"16GFCSMr6_1","T11 FibreChan"
MASKS:ADD:STAN <source>,"32GFCMMr310","T11 FibreChan"
MASKS:ADD:STAN <source>,"32GFCSMr310","T11 FibreChan"

To add Infiniband Masks:

MASKS:ADD:STAN <source>,"IB_EDR25GIn","Infiniband"
MASKS:ADD:STAN <source>,"IB_EDR25GOut","Infiniband"

MASKS:ADD:STAN <source>,"IB_FDR14GIn","Infiniband"
MASKS:ADD:STAN <source>,"IB_FDR14GTX","Infiniband"

MASKS:ADD:STAN <source>,"IB_QDR10GIn","Infiniband"
MASKS:ADD:STAN <source>,"IB_QDR10GOut","Infiniband"

MASKS:ADD:STAN <source>,"IBTA_2G5_Op","Infiniband"
MASKS:ADD:STAN <source>,"IBTA_5G_Op","Infiniband"

To add OUT ITU Masks:

MASKS:ADD:STAN <source>,"100G_OTL4_4","OTU ITU"

MASKS:ADD:STANdard

Description
This command adds a new standard mask measurement. The mask identifier MASK<x> and the subcategory are optional parameters.

Syntax
:MASKS:ADD:STANdard { M[n]{A|B} | MATH[x] | REF[x] },<maskName>,<standardName>[,<standardSubcategory>] [, MASK<x>]

Arguments
{ M[n]{A|B} | MATH[x] |REF[x] } is the source for the mask test

- M[n]{A|B} selects a channel waveform source.
- MATH[x] selects a math waveform source.
- REF[x] selects a reference waveform source.

<maskName> is a string parameter identifying the name of the mask
<standardName> is a string parameter identifying the category of the mask
<standardSubcategory> is an optional string parameter identifying the sub-category of the mask

MASK<x> is the optional mask ID the user wants to create the mask measurement on.

Returns
None

Examples
:MASKS:ADD:STANdard M1A,"10G5BASER","IEEE" shall add a standard mask measurement which may be identified as MASK1. The standard identified with this mask is IEEE and the mask name is 10G5BASER.
:MASKS:ADD:STANdard M2A,"10G5BASER","IEEE",MASK3 shall add a standard mask measurement which is identified as MASK3. The standard identified with this mask is IEEE and the mask name is 10G5BASER.

MASKS:MASK<x>:DELete

Description
This command deletes the mask with specified mask ID MASK<x>. If the mask specified by the mask ID does not exist, then an error shall be reported.

Syntax
MASKS:MASK<x>:DELETE

Arguments
None

Returns
None

Examples
MASKS:MASK7:DELETE shall delete the mask with mask ID MASK7

MASKS:DELete:ALL

Description
This command deletes all the masks added for mask measurement.

Syntax
MASKS:DELETE:ALL

Arguments
None

Returns
None

Examples
MASKS:DELETE:ALL shall delete all the masks preset for mask measurement

MASKS:MASK<x>:SOURce

Description
This command shall set or get the source of mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:SOURce { M[n]{A|B} | MATH[x] | REF[x] }
:MASKS:MASK<x>:SOURce?
Arguments
{ M[n]{A|B} | MATH[x] | REF[x] } is the source for the mask test

- M[n]{A|B} selects a channel waveform source.
- MATH[x] selects a math waveform source.
- REF[x] selects a reference waveform source.

Returns
The quoted string source of the specified mask measurement.

Examples
:MASKS:MASK2:SOURCE M2A shall set the source of mask measurement MASK2 to the source 2A.

:MASKS:MASK1:SOURCE? may return M1A as the source of mask measurement to indicate that the source of the mask measurement is 1A.

MASKS:MASK<x>:DISPlay

Description
This command shall set or get the display state of the mask measurement identified by <x>

Syntax
:MASKS:MASK<x>:DISPlay { ON | OFF | 0 | 1 }
:MASKS:MASK<x>:DISPlay?

Arguments
ON or any other non-zero value turns on the mask measurement display
OFF or 0 turns off the mask measurement display

Returns
0 or 1 indicating state of display of the specified mask measurement as OFF or ON respectively

Examples
:MASKS:MASK2:DISPLAY 1 turns on the display for MASK2
:MASKS:MASK2:DISPLAY? might return 0 indicating that the display for MASK2 is turned off

MASKS:MASK<x>:AFIT

Description
This command shall set or get the horizontal autofit state used to shift the mask horizontally for maximum mask margin of the mask measurement identified by <x>

Syntax
:MASKS:MASK<x>:AFIT { ON | OFF | 0 | 1 }
:MASKS:MASK<x>:AFIT?

Arguments
ON or any other non-zero value starts the mask horizontal autofit computation
OFF or 0 stops the mask horizontal autofit computation

Returns
0 or 1 indicating state of horizontal autofit computation of the specified mask measurement as OFF or ON respectively

Examples
:MASKS:MASK2:AFIT 1 starts horizontal autofit computation for MASK2
:MASKS:MASK2:AFIT? might return 0 indicating that the horizontal autofit computation for MASK2 is stopped

MASKS:MASK<x>:STANdard? (Query only)

Description
This query only command shall return the Standard this mask identified as MASK<x> corresponds to.

Syntax
:MASKS:MASK<x>:STANdard?

Arguments
None

Returns
<string> value indicating the Standard against which the mask is defined

Examples
:MASKS:MASK1:STANdard? may return "IEEE" indicating the IEEE standard to which the mask conforms.

MASKS:MASK<x>:MTESt:ENABle

Description
This command may be used to get or set the margin test state as enable or disable for the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:MTESt:ENABle { ON | OFF | 0 | 1 }
:MASKS:MASK<x>:MTESt:ENABle?

Arguments
ON or any other non-zero value turns on the mask measurement margin test
OFF or 0 turns off the mask measurement margin test

Returns
0 or 1 indicating state of margin test of the specified mask measurement as OFF or ON respectively

Examples
:MASKS:MASK1:MTEST:ENABLE 1 shall enable the mask margin test for mask measurement MASK1.
:MASKS:MASK2:MTEST:ENABLE? may return 0 indicating that the mask margin test for mask measurement MASK2 is disabled.

MASKS:MASK<x>:MTESt:SELection

Description
This command shall allow setting the margin mask test criteria of the mask measurement identified with <x> as one of the following types :
- Manual
- Target Hit Ratio
- Target Hits

Syntax
:MASKS:MASK<x>:MTESt:SELection {MANual | HRATio | HITS]
:MASKS:MASK<x>:MTESt:SELection?

Arguments
- MANual shall set the margin mask test criteria to a margin value given by the user.
- HRATio shall set the margin mask test criteria to a target hit ratio value given by the user.
- HITS shall set the margin mask test criteria to a target hits value given by the user.

Returns
The current margin mask test selection type.

Examples
:MASKS:MASK1:MTEST:SELECTION HRATIO shall set the criteria for margin mask test to use target hit ratio for mask measurement identified as MASK1.
:MASKS:MASK2:MTEST:SELECTION HITS shall set the criteria for margin mask test to use target hits for mask measurement identified as MASK2.
:MASKS:MASK1:MTEST:SELECTION? may return MANUAL indicating the criteria for margin mask test is set by the margin value for mask measurement identified as MASK1.

MASKS:MASK<x>:MTESt:MANual

Description
This command shall get or set the margin mask test target margin value in percentage for the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:MTESt:MANual<NR3>
:MASKS:MASK<x>:MTESt:MANual?

Related Commands

Arguments
<NR3> indicating the Target Margin of the margin mask test in percentage.

Returns
<NR3> indicating the Target Margin of the margin mask test in percentage.

Examples
:MASKS:MASK1:MTEST:MANUAL 30 shall set the margin mask test target margin value to 30% for mask measurement MASK1.
:MASKS:MASK1:MTEST:MANUAL? may return 5 as the margin mask test target margin value for mask measurement MASK1.

MASKS:MASK<x>:MTESt:THITs

Description
This command shall get or set the margin mask test target margin value in percentage for the mask measurement identified by <x>.
Syntax
:MASKS:MASK<x>:MTESt:MANual<NR3>
:MASKS:MASK<x>:MTESt:MANual?

Arguments
<NR3> indicating the Target Margin of the margin mask test in percentage.

Returns
<NR3> indicating the Target Margin of the margin mask test in percentage.

Examples

:MASKS:MASK1:MTEST:MANUAL 30 shall set the margin mask test target margin value to 30% for mask measurement MASK1.
:MASKS:MASK1:MTEST:MANUAL? may return 5 as the margin mask test target margin value for mask measurement MASK1.

MASKS:MASK<x>:MTESt:THRatio? (Query only)

Description
This query only command shall return the margin mask test target hit ratio as a ratio of population value for the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:MTESt:THRatio?

Arguments
None

Returns
<NR3> indicating the Target Hit Ratio of the margin mask test.

Examples
:MASKS:MASK1:MTEST:THRATIO? may return 12e-6 as the margin mask test hit ratio value for mask measurement MASK1.

MASKS:MASK<x>:PFTest:ENABle

Description
This command may be used to get or set the pass/fail test state as enable or disable for the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:PFTest:ENABle { ON | OFF | 0 | 1 }
:MASKS:MASK<x>:PFTest:ENABle?

Arguments
ON or any other non-zero value turns on the mask measurement pass/fail test
OFF or 0 turns off the mask measurement pass/fail test

Returns
0 or 1 indicating state of pass/fail test of the specified mask measurement as OFF or ON respectively

Examples
:MASKS:MASK1:PFTEST:ENABLE 1 shall enable the mask pass/fail test for mask measurement MASK1.
:MASKS:MASK2:PFTEST:ENABLE? may return 0 indicating that the mask pass/fail test for mask measurement MASK2 is disabled.

MASKS:MASK<x>:PFTest:TMARgin

Description

This command sets or queries the mask pass/fail test margin value in percentage for the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:PFTest:TMARgin <NR3>
:MASKS:MASK<x>:PFTest:TMARgin?

Arguments
<NR3> indicates the Target Margin of the mask pass/fail test in percentage.

Returns
<NR3> indicates the Target Margin of the mask pass/fail test in percentage.

Examples
:MASKS:MASK1:PFTEST:TMARGIN 17 sets the mask pass/fail test target margin value to 17% for mask measurement MASK1.
:MASKS:MASK1:PFTEST:TMARGIN? may return 7, indicating 7% as the pass/fail margin criteria for mask measurement MASK1.

MASKS:MASK<x>:PFTest:THITs

Description
This command shall get or set the mask pass/fail test target hits value for the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:PFTest:THITs <NR3>
:MASKS:MASK<x>:PFTest:THITs?

Related Commands

Arguments
<NR3> indicating the Target Hits of the mask pass/fail test.

Returns
<NR3> indicating the Target Hits of the mask pass/fail test.

Examples
:MASKS:MASK1:PFTEST:THITS 29 shall set the mask pass/fail test target hits value to 29 hits for mask measurement MASK1.
:MASKS:MASK2:PFTEST:THITS? may return 137 as the mask pass/fail test hits value for mask measurement MASK2 indicating 137 hits as the pass/fail hits criteria.

MASKS:MASK<x>:PFTest:THRatio

Description
This query only command shall return the mask pass/fail test target hit ratio as a ratio of population value for the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:PFTest:THRatio?

Arguments
None

Returns
<NR3> indicating the Target Hit Ratio of the mask pass/fail test.

Examples
:MASKS:MASK2:PFTEST:THRatio? may return 13.7e-12 as the mask pass/fail test target hit ratio value for mask measurement MASK2 indicating 13.7e-12 hit ratio as the pass/fail criteria.

MASKS:MASK<x>:RESult:STATus? (Query only)

Description
This query only command shall return the result status of the mask measurement identified by <x>

Syntax
:MASKS:MASK<x>:RESult:STATus?

Arguments
None

Returns
0 or 1 indicating state of result of the mask measurement test Fail or PASS respectively

Examples
:MASKS:MASK1:RESULT:STATUS? 1 indicating a Pass result for mask measurement identified as MASK1.

MASKS:MASK<x>:RESult:ATTRibutes

Description
This query only command shall return a comma separated list of result attributes available for the mask measurement identified by <x>

Syntax
:MASKS:MASK<x>:RESult:ATTRibutes?

Arguments
None

Returns
<string> the comma separated list of result attributes.

Examples
:MASKS:MASK1:RESULT:ATTRIBUTES? may return "Mask Region 1 Hits,Mask Region 2 Hits,Mask Region 3 Hits,Mask Total Hits,Mask Hit Ratio" as the list of result attributes when margin test is turned for mask measurement MASK1

:MASKS:MASK2:RESULT:ATTRIBUTES? may return "Mask Region 1 Hits,Mask Region 2 Hits,Mask Region 3 Hits,Mask Total Hits,Mask Hit Ratio,Margin Mask Region 1 Hits,Margin Mask Region 2 Hits,Margin Mask Region 3 Hits,Margin Mask Total Hits,Margin Mask Hit Ratio" as the list of result attributes when margin test is turned on for mask measurement MASK2

MASKS:MASK<x>:RESult:VALue? (Query only)

Description
This query only command shall return the result of the parameter identified by <attribute> of the mask measurement identified by <x>.

Syntax
:MASKS:MASK<x>:RESult:VALue? <attribute>

Arguments
<attribute> string identifier available from :MASKS:MASK<x>:RESult:ATTRibutes? query

Returns
<NR3> the value associated with the <attribute>

Examples
:MASKS:MASK1:RESULT:VALUE? "Mask Hit Ratio" may return 3e-9 indicating 3e-9 as the current hit ratio of the mask test for the mask measurement identified as MASK1.
:MASKS:MASK2:RESULT:VALUE? "Margin Mask Total Hits" may return 37 indicating 37 as the total hits for the margin mask for the mask measurement identified as MASK2.

MASKS:MASK<x>:SAVE

Description
This command saves the mask with specified mask ID MASK<x> to the file specified by <filename>.
If the mask specified by the mask ID does not exist, then an error shall be reported.

Syntax
MASKS:MASK<x>:SAVE <filename>

Arguments
<filename> is the name of the file along with the full path that the mask definition shall be saved to

Returns
None

Examples
MASKS:MASK1:SAVE "C:\Masks\100GBaseLR4.msk" shall save the mask used in the mask measurement MASK1 to file 100GBaseLR4.msk at location C:\Masks

Miscellaneous Command Group

AUTOSet (Command only)

Description
This command starts the autoset process for the selected waveform. Autoset adjusts vertical settings for the selected waveform to provide ideal display of the waveform.

Note: Autoset only adjusts vertical settings.

Syntax

:AUTOSet [EXECute]

Related Commands
:SELect:CONTROl

Arguments
[EXECute] This is an optional argument that autosets the selected waveform.

Examples
AUTOSET EXECUTE performs an autoset on the currently selected waveform.

Limitations
Autoset can only be run when there is an acquired waveform present for the specified live channel.
Autoset cannot be run while connected to a Virtual Instrument

*IDN? (Query Only)[WD3]

Description
This query-only command returns the instrument identification code.

Syntax
*IDN?

Returns
Instrument manufacturer, model, serial number, and firmware version number.

<field1>,<field2>,<field3>,<field4>
where
<field1> = Manufacturer
<field2> = Model
<field3> = Serial Number (ASCII character 0 if not available)
<field4> = Firmware level or equivalent (ASCII character 0 if not available)

Examples
*IDN? might return TEKTRONIX,TSOVu,0,CF:91.1CT FV:1.2.3.4 indicating the instrument manufacturer, model, serial number, and firmware version number. Note that serial number will be '0' when no instruments are attached.

:HEADer

Description
This command sets or queries the Response Header Enable State that causes the instrument to either include or omit headers on query responses. This command does not affect IEEE Std 488.2-1987 Common Commands (those starting with an asterisk); these commands never return headers.

Syntax
HEADer?
HEADer { 0 | 1 | ON | OFF}
HDR?
HDR { 0 | 1 | ON | OFF }

Arguments
0 or OFF disables the Response header state
1 or ON enables the Response header state

Returns
0 if the the Response headers are disabled
1 if the Response headers are enabled

Examples
HEADER 0 sets the response header state to disabled and causes the instrument to omit headers from query responses.
HEADER? might return 0 if response header state is set to disabled, indicating the queries will return responses with headers.

:VERBose

Description
This command sets or queries the verbose state that controls the length of keywords on query responses. Keywords can be both headers and arguments. This command does not affect IEEE Std 488.2-1987 Common Commands (those starting with an asterisk).

Syntax
VERBose?
VERBose { 0 | 1 | ON | OFF }

Arguments
ON or 1 sets the verbose state true, which returns full-length keywords for applicable setting queries.
OFF or 0 sets the verbose state false, which returns minimum-length keywords for applicable setting queries.

Returns
0 indicating that the verbose state is set to false, and minimum-length keywords will be returned
1 indicating that the verbose state is set to true, and the full-length keywords will be returned

Examples
VERBOSE ON sets the verbose state true, which returns full-length keywords for queries.
VERBOSE? might return VERB 0 indicating that the minimum-length keyword for the query will be returned

Save and Recall Command Group

You 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 saved setting, the instrument restores itself to the state that it was in when you originally saved that setting.

RECAll:WAVEform

Description
The user shall use this PI command to recall the waveform. This command (no query form) recalls a stored waveform into a reference location.

Syntax
:RECAll:WAVEform <file path>,REF[n]

Arguments

- <filepath> 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 recall the waveform from the default directory. The <filename> can be a Windows long file name. Do not use wild card characters.
- REF<x> is the location in internal reference memory to which the waveform is recalled.

Examples
:RECALL:WAVEFORM "TEK000.WFM",REF2 recalls the waveform file Tek000.wfm from the default TekScope save location to Ref2.

SAVe:WAVEform

Description
The user shall use this PI command to save the waveform.
This command (no query form) stores a waveform to a file. You must specify a waveform and a destination file path. Sending this command is equivalent to selecting Save As... from the File menu, selecting the Waveform tab, and then selecting a waveform and destination in the Save Waveform dialog box.

Syntax
:SAVe:WAVEform { M[n]{A|B}| MATH[x] | REF[x]}, <filepath>

Arguments
- M[n]{A|B} selects a channel waveform to save. The range for n is 1 through the number of connected modules.
- MATH[x] selects a math waveform to save. REF[x] selects a reference waveform to save.
- <file path> is the location to which the waveform will be saved.
- The <file path> is a quoted string that defines the file name and path. Input the file path using the form <drive>/<dir>/<filename>. The <filename> can be a Windows long file name. Do not use wild card characters

Examples
:SAVE:WAVEFORM M1B,"TEK00000.WFM" saves the module 1 channel B waveform displayed to the file TEK00000.WFM in the default directory and on the default drive

DELEte:WAVEform

Description
The user shall use this PI command to delete one or all of the stored reference waveforms. This command (no query form) deletes one of the stored reference waveforms from memory.

NOTE: If a reference waveform is deleted and it is the source for one or more measurements, those measurements will also be deleted.

Syntax
:DELEte:WAVEform REF<x>

Arguments
REF<x> specifies one of the reference memory locations.

Examples
:DELETE:WAVEFORM REF2 removes the waveform stored at REF2.

SAVE:REPort

Description
This command (no query form) saves the measurement report into the specified memory location. Sending this is equivalent to selecting the Report Tab in the Save As panel and then clicking the Save button.

Syntax
SAVe:REPort <file path>[,<{FREPort | SUMMary}>]

Arguments
1st argument
<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 write the file to the current directory. The <filename> can be a Windows long file name. Do not use wild card characters. If the ".tss" extension is not provided, it will be appended to the filename. To find out what can be saved, please refer to the Save Report requirement specification.

2nd argument
FREPort specifies that the comprehensive report will be saved.
SUMMary specifies that the summary report will be saved.

Examples
SAVE:REPORT "C:\MY DOCUMENTS\TESTS\UI\DATA\ReportTEST.html", FREPort saves the comprehensive report to the file ReportTEST.html in the Data directory on the C drive.
SAVE:REPORT "C:\MY DOCUMENTS\TESTS\UI\DATA\ReportTEST.html", SUMMary saves summary reports in the Data directory on the C drive.

SAVe:SETUp

Description
This command (no query form) saves the current instrument setup into the specified memory location. Sending this is equivalent to selecting Save Setup in the File menu.

Syntax
SAVe:SETUp <file path>[,<{ON|OFF|1|0}>]

Related Commands
RECAll:SETUp

Arguments
<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 write the file to the current directory. The <filename> can be a Windows long file name. Do not use wild card characters. If the ".tss" extension is not provided, it will be appended to the filename. To find out what can be saved, please refer to the Save Setup requirement specification.

[,<{ON|OFF|1|0}>]
- ON specifies that save setup includes reference waveform.
- OFF specifies that save setup does not include reference waveform.
- 1 enables reference waveform saving.
- 0 disables reference waveform saving.

It is optional.

Examples
SAVE:SETUP "C:\MY DOCUMENTS\TESTS\UI\DATA\SETUPTEST.SET" saves the current instrument setup to the file Setuptest.SET in the Data directory on the C drive.
SAVE:SETUP "C:\MY DOCUMENTS\TESTS\UI\DATA\SETUPTEST_REF.SET", ON saves the current instrument setup with all references included to the file Setuptest_ref.SET in the Data directory on the C drive.

RECAll:SETUp

Description
This command (no query form) restores a stored or factory default setup from a Windows file.

Syntax
RECAll:SETUp {FACtory | <file path>}

Related Commands
SAVe:SETUp

Arguments
- FACtory selects the factory setup.
- <file path> is the location from which the setup will be recalled. The <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. Do not use wild card characters.

Examples
- RECALL:SETUP FACTORY recalls (and makes current) the instrument setup to its factory defaults.
- RECALL:SETUP "TEK00000.SET" recalls the instrument setup from the file TEK00000.SET in the default directory and on the default drive.

SAVe:SESSion

Description
Saves the state of the instrument and all acquired waveform data to a zipped session file. This enables the user to move analysis activities offline.

Syntax
:SAVe:SESSion <file path>

Arguments
<file path> is a quoted string that defines the file name and path of the location to save the session archive to. 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 file to the current directory. The <filename> can be a Windows long file name. Do not use wild card characters. If the ".tss" extension is not provided, it will be appended to the filename.

Examples
- :SAVE:SESSION "C:\MY_DOCUMENTS\TESTS\DATA\TEST1.TSS" saves the current instrument state with all waveform sources to the file Test1.tss in the Data directory on the C drive.
- :SAVE:SESSION "C:\MY_DOCUMENTS\TESTS\DATA\TEST2" saves the current instrument state with all waveform sources to the file Test2.tss in the Data directory on the C drive.

RECAll:SESSion

Description
This command (no query form) restores a stored session from a Windows file.

Syntax
:RECAll:SESSion <file path>

Related Commands
:SAVe:SESSion

Arguments
<file path> is the location from which the session will be recalled. The <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. Do not use wild card characters.

Examples
:RECALL:SESSION "TEK00000.TSS" recalls the instrument session from the file TEK00000.TSS in the default directory and on the default drive.

SAVe:SCAPture (Command only)

Description
This command (no query form) saves the main window screen capture into the specified memory location. Sending this command is equivalent to selecting Screen Capture Tab in Save As Panel and clicking on Save button.

Syntax
:SAVe:SCAPture <filepath>

Arguments
<file path> is the file name and path of the location to which the captured screen will be saved. 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 file to the current directory. The <filename> can be a Windows long file name. Do not use wild card characters. To find out supported file format, please refer to the Screen Capture Tab requirement specification.

Examples
SAVe:SCAPture "C:\MY_DOCUMENTS\TESTS\DATA\TEST1.png" saves main window screen capture to the file Test1.png in the Data directory on the C drive.
SAVe:SCAPture "C:\MY_DOCUMENTS\TESTS\DATA\TEST1.bmp" saves main window screen capture to the file Test1.bmp in the Data directory on the C drive.
SAVe:SCAPture "C:\MY_DOCUMENTS\TESTS\DATA\TEST1.jpg" saves main window screen capture to the file Test1.jpg in the Data directory on the C drive.

Status and Error Command group

You 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.

*ESR? (Query Only)

Description
This is a query only command that returns the contents of the Standard Event Status Register (SESR). This query also clears the SESR, since reading the SESR clears it. For a more detailed discussion of the use of these registers, see Registers.

Syntax
*ESR?

Examples
*ESR? might return the value 213, showing that the SESR contains binary 11010101.

:ALLev?

Description
This is a query only command that returns a description of the error that has occurred.

Syntax
:ALLev?

Returns
The event code and message in the following format:<Event Code>,<QString>[<EventCode>,<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> strings combined. The command string is right justified.

Examples
*ALLev? might return *ALLev? 200,"Execution error"

*OPC

Description

This command generates the operation complete message in the Standard Event Status Register (SESR) when all pending operations finish. The *OPC? query places the ASCII character "1" into the output queue when all pending operations are finished. 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.

Syntax
*OPC
*OPC?

Returns
A single <NR1> value.

Examples
*OPC? returns 1 to indicate that the last issued overlapping command is finished.

*RST (No Query Form)

Description
This command (no query form) resets TSOVu and the connected instruments to the factory default settings.

The *RST command has the following effect:
- All defaultable application settings are reset to their default values
- All reference waveforms are deleted
- All measurements are deleted
- Acquisition is stopped and acquisition count and averaging are reset
- Connected instruments **ARE NOT** disconnected

The *RST command does not alter the following:
- Calibration data that affect device specifications
- The Output Queue
- The Service Request Enable Register setting
- The Standard Event Status Enable Register setting
- The Power-on status clear flag setting

Syntax
*RST

Examples
*RST resets TSOVu and the connected instruments' settings to factory defaults.

*ESE

Description
This command sets or queries the bits in the Event Status Enable Register (ESER). The ESER prevents events from being reported to the Status Byte Register (SBR). For a more detailed discussion of the use of these registers, see Registers.

Syntax
- *ESE?
- *ESE <NR1>

Arguments
NR1 specifies the binary bits of the ESER according to this value, which ranges from 0 through 255.

Returns
The power-on default for ESER is 0 if *PSC is 1. If *PSC is 0, the ESER maintains its value through a power cycle.

Examples
- *ESE 209 sets the ESER to binary 11010001, which enables the PON, URQ, EXE, and OPC bits.
- *ESE? might return the string *ESE 186, indicating that the ESER contains the binary value 10111010.

*SRE

Description
The *SRE (Service Request Enable) command sets or queries the bits in the Service Request Enable Register (SRER). For more information, refer to Registers.

Syntax
- *SRE?
- *SRE <NR1>

Arguments
- NR1 is a value in the range from 0 to 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 its value throughout a power cycle.

Returns
A value from 0 to 255 representing the binary bits of the SRER.

Examples
- *SRE 48 sets the bits in the SRER to 00110000 binary.
- *SRE? returns a value of 32, showing that the bits in the SRER have the binary value 00100000.

*CLS

Description
This command (no query form) clears the following status data structures of the instrument:
- Event Queue
- Standard Event Status Register (SESR)
- Status Byte Register (except the MAV bit; see below)

If the *CLS command immediately follows an <EOI>, the Output Queue and MAV bit (Status Byte Register bit 4) are also cleared. MAV indicates 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 hardcopy output or conditional acquisition operation is still being processed when the *CLS command is executed.

Syntax
*CLS

Examples
*CLS clears the instrument status data structures.

*STB? (Query Only)

Description
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.

Syntax
*STB?

Related Commands
*CLS
DESE
*ESE
*ESR?
EVENT?
EVMsg?
FACtory
*SRE

Returns
an integer between 0-255 representing the contents of the Status Byte Register (SBR)

Examples
*STB? might return 96 showing that the SBR contains the binary value 01100000.

STATus:OPERation:CONDition? (Query Only)

Description
This command returns the contents of the Operation Condition Register (OCR).

Syntax
STATus:OPERation:CONDition?

Returns
A single <NR1> value showing the contents of the OCR.

Examples
STATUS:OPERATION:CONDITION? might return 0, showing that the bits in the OCR have the binary value 0000000000000000.

STATus:OPERation:ENABle

Description
This command sets or returns the mask for the Operation Enable Register.

Syntax
STATus:OPERation:ENABle <NR1>
STATus:OPERation:ENABle?

Arguments
A single <NR1> value.
Range: 0 to 65535

Returns
A single <NR1> value.

Examples
STATUS:OPERATION:ENABLE 1 enables the Calibrating bit.
STATUS:OPERATION:ENABLE? might return 1, showing that the bits in the OENR have the binary value 00000000 00000001, which means that the Calibrating bit is valid.

Limitations
The most-significant bit cannot be set true.

STATus:OPERation[:EVENt]? (Query Only)

Description
This command returns the contents of the Operation Event Register (OEVR).
Reading the OEVR clears it.

Syntax
STATus:OPERation[:EVENt]?

Returns
A single <NR1> value showing the contents of the OEVR.

Examples
STATUS:OPERATION:EVENT? might return 1, showing that the bits in the OEVR have the binary value 00000000 00000001, which means that the CALibrating bit is set.

Limitations
<List of noted limitations in the command's functionality>

STATus:OPERation:NTRansition

Description
This command sets or returns the negative transition filter value of the Operation Transition Register (OTR).

Syntax
STATus:OPERation:NTRansition <bit_value>
STATus:OPERation:NTRansition?

Arguments
<bit_value> ::= <NR1> is the negative transition filter value.
Range: 0 to 65535

Returns
A single <NR1> value showing the contents of the OTR.

Examples
STATUS:OPERATION:NTRANSITION 17 sets the negative transition filter value to 17.
STATUS:OPERATION:NTRANSITION? might return 17.

Limitations
The most-significant bit cannot be set true.

STATus:OPERation:PTRansition

Description
This command sets or returns the positive transition filter value of the Operation Transition Register (OTR).

Syntax

STATus:OPERation:PTRansition <bit_value>
STATus:OPERation:PTRansition?

Arguments
<bit_value> ::= <NR1> is the positive transition filter value.
Range: 0 to 65535.

Returns
A single <NR1> value showing the contents of the OTR.

Examples
STATUS:OPERATION:PTRANSITION 0 sets the positive transition filter value to 0.
STATUS:OPERATION:PTRANSITION? might return 0.

Limitations
The most-significant bit cannot be set true.

STATus:PRESet

Description
This command resets the SCPI Operation and Questionable Enable and Transition registers to zero.

Syntax
STATus:PRESet

Examples
STATUS:PRESET sets the Operation Enable Register (OENR), Operation Transition Register (OTR), Questionable Enable Register (QENR), and Questionable Transition Register (QTR) to zero.

System Command Group

SYSTem:CONNect <Hostname>

Description
This command is used to make a connection with an instrument remotely by Hostname or IP address

Syntax
:SYSTem:CONNect <Hostname>

Related Commands
:SYSTem:DISCONNect

Arguments
<Hostname> is the quoted host name or IP address of the instrument to make a connection with.

Examples
:SYSTEM:CONNECT "0.0.0.0" will make the connection with the instrument "0.0.0.0" if it is available.

SYSTem:CONNect:STATus? [<Hostname>] (Query Only)

Description

Returns the connection status.

Syntax
:SYSTem:CONNect:STATus? [<Hostname>]

Related Commands
:SYSTem:CONNect <Hostname>
:SYSTem:DISCONNect

Arguments
[<Hostname>] is the quoted host name or IP address of an instrument

Returns
0 if instrument is not connected, or is having a connection issue
1 if instrument is connected

Examples

- :SYSTEM:CONNECT:STATUS? might return SYSTEM:CONNECT:STATUS 1 indicating an instrument is currently connected to the scope app with no connection issues
- :SYSTEM:CONNECT:STATUS? "0.1.2.3" might return SYSTEM:CONNECT:STATUS "0.1.2.3",0 indicating the instrument "0.1.2.3" is currently having connection issues with the scope app.
- :SYSTEM:CONNECT:STATUS? "0.0.0.0" might report an error and return SYSTEM:CONNECT:STATUS "0.0.0.0",0 indicating that the instrument "0.0.0.0" is not a source recognized from the instruments list and is not connected to the scope app.

SYSTem:DISConnect

Description
This command is used to disconnect a connected instrument.
Note: Disconnecting an instrument will delete any measurements depending on any of the instrument's live channels as sources.

Syntax
:SYSTem:DISConnect

Related Commands
:SYSTem:CONNect <Hostname>

Examples
:SYSTEM:DISCONNECT will disconnect the instrument

SYSTem:PROPerties:MODel? (Query Only)

Description
Returns the given instrument's model number.

Syntax
:SYSTem:PROPerties:MODel? (Query Only)

Returns
The model number of the specified instrument.

Examples

:SYSTEM:PROPERTIES:MODEL? might return SYSTEM:PROPERTIES:MODEL "TSO5" indicating that the instrument's model number is "TSO5".

SYSTem:PROPerties:SERial? (Query Only)

Description
Returns the given instrument's serial number.

Syntax
:SYSTem:PROPerties:SERial? (Query Only)

Returns
The serial number of the specified instrument.

Examples
:SYSTEM:PROPERTIES:SERIAL? might return SYSTEM:PROPERTIES:SERIAL "B0000327" indicating that the instrument's serial number is "B0000327".

SYSTem:PROPerties:FVERsion? (Query Only)

Description
Returns the given instruments software version.

Syntax
:SYSTem:PROPerties:FVERsion? (Query Only)

Returns
The software version of the specified instrument.

Examples
:SYSTEM:PROPERTIES:FVERSION? might return SYSTEM:PROPERTIES:FVERSION "1.0.10.1" indicating that the instrument's software version is "1.0.10.1"

SYSTem:PROPerties:M<x>[A|B]:SERial? (Query Only)

Description
Returns the given module's serial number. The module number and channel are specified by <X> [A|B].

Syntax
:SYSTem:PROPerties:M<x>[A|B]:SERial? (Query Only)

Returns
The serial number of the specified module.

Examples
:SYSTEM:PROPERTIES:M1A:SERIAL? might return SYSTEM:PROPERTIES:M1A:SERIAL "A001" indicating that the serial number for Module 1 is "A001"

SYSTem:PROPerties:M<x>[A|B]:MODel? (Query Only)

Description

Returns the given module's model number. The module number is specified by <X>[A|B].

Syntax
:SYSTem:PROPerties:M<x>[A|B]:MODel? (Query Only)

Returns
The serial number of the specified module.

Examples
:SYSTEM:PROPERTIES:M1A:MODEL? might return SYSTEM:PROPERTIES:M1A:MODEL "0E1" indicating that module 1 has a model number of "0E1"

SYSTem:PROPerties:M<x>[A|B]:FVERsion? (Query Only)

Description

Returns the given module's firmware version. The module number and channel are specified by <X> [A|B].

Syntax
:SYSTem:PROPerties:M<x>[A|B]:FVERsion? (Query Only)

Returns
The firmware version of the specified module.

Examples
- :SYSTEM:PROPERTIES:M1A:FVERSION? might return SYSTEM:PROPERTIES:M1A:FVERSION "1.0" indicating that the firmware version for Module 1 is "1.0"

SYSTem:PROPerties:HOSTname? (Query only)

Description
Returns the given instrument's hostname or IP address.

Syntax
:SYSTem:PROPerties:HOSTname? (Query only)

Returns
The hostname of the specified instrument.

Examples
:SYSTEM:PROPERTIES:HOSTNAME? might return SYSTEM:PROPERTIES:HOSTNAME "0.0.0.0" indicating that the instrument's hostname is "0.0.0.0".

SYSTem:PROPerties:MODUle:COUNt? (Query only)

Description
Returns the count number of the connected instrument's modules.

Syntax
:SYSTem:PROPerties:MODUle:COUNt? (Query only)

Returns
The count of the number of modules of the connected instrument.

Examples

- :SYSTEM:PROPERTIES:MODULE:COUNT? might return SYSTEM:PROPERTIES:MODULE:COUNT 2 indicating that the connected instrument has 2 connected modules.
- :SYSTEM:PROPERTIES:MODULE:COUNT? might return SYSTEM:PROPERTIES:MODULE:COUNT 0 indicating that the connected instrument has 0 connected modules.

SYSTem:PROPerties:M<x>:COUNt? (Query only)

Description
Returns the count number of the connected module's channels.

Syntax
:SYSTem:PROPerties:M<x>:COUNt? (Query only)

Returns
The count of the number of modules of the connected instrument.

Examples

- :SYSTEM:PROPERTIES:M1:COUNT? might return SYSTEM:PROPERTIES:M1:COUNT 2 indicating that the connected module 1 has 2 channels.
- :SYSTEM:PROPERTIES:M1:COUNT? might return SYSTEM:PROPERTIES:M1:COUNT 0 indicating that there is no module 1 connected.

SYSTem:REConnect

Description

Manually reconnect with an instrument when connection issues occur.
Note: Problems with instrument connection are indicated by SYSTEM:CONNECT:STATUS 0 with no errors reported.

Syntax
:SYSTem:REConnect

Related Commands
:SYSTem:CONNect <hostname>
:SYSTem:DISConnect
:SYSTem:CONNect:STATus?

Examples

- After the command :SYSTEM:RECONNECT, the query :SYSTEM:CONNECT:STATUS? might return :SYSTEM:CONNECT:STATUS 1 indicating that reconnecting to the instrument has succeeded, and the instrument is connected.
- After the command :SYSTEM:RECONNECT, the query :SYSTEM:CONNECT:STATUS? might return :SYSTEM:CONNECT:STATUS 0 indicating that the attempt to reconnecting to the instrument has failed, and the instrument connection issues persist.

Trigger Command Group

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

TRIGger:SOUrce

Description
This command sets or queries the instrument trigger source, which specifies the signal that starts (triggers) signal acquisition.

Syntax
:TRIGger:SOUrce {CLKPre | FREerun }
:TRIGger:SOUrce?

Arguments
CLKPre sets the instrument to trigger on an external signal connected to the CLOCK INPUT/PRESCALE TRIGGER input connector on the instrument front panel. The input coupling is AC. Triggering is guaranteed for clocks up to 15 GHz (and will typically work up to 20 GHz).
FREerun sets the instrument to disable synchronous triggered acquisition; the instrument randomly triggers to acquire data.

Returns
The current trigger source, CLKPRE, or FREERUN

Examples
- :TRIGGER:SOURCE CLKPRE sets the instrument to trigger on an external signal connected to the CLOCK INPUT/PRESCALE TRIGGER input connector on the instrument front panel.
- :TRIGGER:SOURCE? might return FREERUN, indicating that the trigger source is currently set to Free Run with an Untriggered Phase Reference

Notes
When trigger source is configured to FREERUN, if horizontal pattern sync is not OFF, pattern sync shall be turned OFF and HORIZONTAL:PSYNC? shall return HORIZONTAL:PSYNC 0

TRIGger:CFREquency (Query Only)

Description
This command queries the trigger clock frequency.

Syntax
:TRIGger:CFREquency?

Returns
The current trigger clock frequency

Examples
:TRIGGER:CFREQUENCY? might return TRIGGER:CFREQUENCY 16.000000000000E+9, indicating that the internal clock rate is set to 16 GHz.

TRIGger:PSYNc:PLENgth

Description
It is better to use new command HORizontal:PLENgth. The purpose of this command is for backward compatibility to DSA8300.

This command sets or queries the pattern length (<NR1>) of the PatternSync Trigger portion of the trigger system. The pattern length must be in the range of 2 to 223 (8,388,608)

Syntax
:TRIGger:PSYNc:PLENgth <NR1>
:TRIGger:PSYNc:PLENgth?

Arguments
<NR1> sets the pattern length in bits.

Examples
- :TRIGGER:PSYNC:PLENGTH 127 sets the pattern sync pattern length to 127 bits.
- :TRIGGER:PSYNC:PLENGTH? might return TRIGGER:PSYNC:PLENGTH 32767, indicating that the pattern length is 32,767 bits.

TRIGger:PSYNc:DATARate

Description
It is better to use new command HORizontal:SRATe. The purpose of this command is for backward compatibility to DSA8300.

This command sets or queries the data rate (<NR3>) of the PatternSync portion of the trigger system.

Syntax
:TRIGger:PSYNc:DATARate <NR3>
:TRIGger:PSYNc:DATARate?

Arguments
<NR3> sets the data rate in bits per second.

Examples
- :TRIGGER:PSYNC:DATARATE 10.3125E9 sets the data rate to 10.3125 Gb/s.
- :TRIGGER:PSYNC:DATARATE? might return TRIGGER:PSYNC:DATARATE 9.95328000E9, indicating that the data rate is 9.95328 Gb/s.

TRIGger:PSYNc:DCRAtio

Description
It is better to use new command HORizontal:DCRAtio. The purpose of this command is for backward compatibility to DSA8300.

This command sets or queries the data-to-clock ratio (<data rate>,<clock rate>) of the PatternSync Trigger portion of the trigger system. The first integer (NR1) value represents the data rate and the second integer (NR1) value represents the clock rate.

Syntax
:TRIGger:PSYNc:DCRAtio <NR1>,<NR1>
:TRIGger:PSYNc:DCRAtio?

Arguments
<NR1> (the first argument) is an integer value (NR1) that sets the data rate.
<NR1> (the second argument) is an integer value (NR1) that sets the clock rate.

The valid Data Rate:Clock Rate ratios are

1:1
2:1
4:1
8:1
16:1
32:1

Examples

- :TRIGGER:PSYNC:DCRATIO 2,1 sets the data-to-clock ratio as 2:1.
- :TRIGGER:PSYNC:DCRATIO? might return TRIGGER:PSYNC:DCRATIO 1,4 indicating that the current data-to-clock ratio is 1:4.

Vertical Command Group

You 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.

SELect:CONTROl

Description
This command sets or queries the currently selected waveform. Use this command to specify the selected waveform for any operations that only affect the currently selected waveform.

Syntax
SELect:CONTROl { M[n]{A|B} | MATH<x> | REF<x> }
SELect:CONTROl?

Arguments
{ M[n]{A\B} | MATH<x> | REF<x> } specifies which waveform and channel to set as currently selected.

Returns
The name of the currently selected waveform.

Examples

- SELECT:CONTROl REF2 sets the Ref2 waveform as the currently selected waveform.
- SELECT:CONTROl? might return "M2B", indicating that channel B on module 2 is currently selected.
- SELECT:CONTROl MATH2 sets the Math2 waveform as the currently selected waveform.

REF[x]:POSition

Description
The user shall use this PI command to set or query the vertical position of the specified reference waveform. The reference waveform is specified by x. This is the equivalent to specifying the Vertical Position in the Ref Configuration Menu (right-click property of the Ref Badge in the Settings Bar at the bottom of the user interface).

Syntax
REF[x]:POSition <NR3>
REF[x]:POSition?

Arguments
<NR3> is the desired position, in divisions from the center graticule. The range is ±1000 divisions.

Returns
The vertical position of the specified reference waveform.

Examples

- REF2:POSITION 1.2E+000 positions the Reference 2 waveform 1.2 divisions above the center of the display.
- REF1:POSITION? Might return REF1:POSITION 1.2000000000, indicating that the Reference 1 waveform is positioned 1.2 divisions above the center of the display.

REF[x]:SCALe

Description
The user shall use this PI command to set or query the vertical scale of the specified reference waveform.
The reference waveform is specified by x. This is the equivalent to specifying the Vertical Scale in the Ref Configuration Menu (right-click property of the Ref Badge in the Settings Bar at the bottom of the user interface). Increasing the Scale causes the waveform to be displayed smaller. Decreasing the scale causes the waveform to be displayed larger. For reference waveforms, this setting controls the display only, graphically scaling these waveforms and having no effect on the acquisition hardware.

Syntax
REF[x]:SCALe <NR3>
REF[x]:SCALe?

Arguments
<NR3> is the vertical reference scale in units per division.

Returns
It returns the vertical scales of the specified reference waveform.

Examples

- REF4:SCALE 1.0E-02 sets the Reference 4 Waveform scale to 10 mV per division.
- REF1:SCALE? Might return REF1:SCALE 10.0000000000E-3, indication that the current volts per division setting of the Reference 1 Waveform is 10 mV per division.

REF[x]:WFMLabel

Description
The user shall use this PI command to set or query the label associated with the reference waveform specified.

Syntax
:REF[x]:WFMLabel <Qstring>
:REF[x]:WFMLabel?

Arguments
<Qstring> sets the label for the reference waveform.

Returns
It returns the label for the specified waveform.

Examples

- :REF1:WFMLABEL “MY REF1 WAVEFORM” sets the label for REF1 to be “My REF1 waveform”, which is the label displayed with the waveform when it is displayed on screen.

- :REF1:WFMLABEL? Might return REF1:WFMLABEL “MY REF1 WAVEFORM”, indicating that the label for the REF1 waveform is set to “My REF1 waveform”.

REF[x]:UNIts

Description
The user shall use this PI command to set or query the vertical scale units associated with the reference waveform specified.
Group: Vertical

Syntax
:REF[x]:UNIts <Qstring>
:REF[x]:UNIts?

Arguments
<Qstring> sets the vertical scale units for the reference waveform.

Returns
It returns the unit of the specified waveform.

Examples
- :REF1:UNITS “W” sets the vertical scale units for the REF1 waveform to be “W”.
- :REF1:UNITS? Might return REF1:UNIT “V”, indicating that the vertical scale units for the REF1 waveform is set to “V”.

DISPlay:REF[x]

Description
The user shall use this PI command to set or query whether the specified reference waveform is displayed.
NOTE: You should define a reference waveform before turning the waveform on.

Syntax
:DISPlay:REF[x] { ON | OFF | 0 | 1 }
:DISPlay:REF[x]?

Arguments
- ON displays the specified reference waveform.
- OFF turns off the display of the specified reference waveform.
- NR1 set to 0 turns off the display of the specified reference waveform; any other value displays the specified reference waveform.

Returns
It returns the display status of the waveform.

Examples
- :DISPlay:REF1 0: it turns off the REF1 display.
- :DISPlay:REF1?: it returns the DISPlay:REF1 0, the waveform display status.

REF[x]:RECordlength?

Description

The user shall use this PI command to query the vertical scale of the specified reference waveform. This is a query only PI command.

Syntax
:REF[x]:RECordlength?

Returns
It returns the record length of the specified waveform.

Examples
:REF1:REC? Might return REF1:RECORDLENGTH 120000 indication that
the current record length value of the Reference 1 Waveform is 120000.

:REF[X]:WFMLabel:XPOS

Description
This command sets or queries the X-position of the specified reference waveform label. The reference waveform is specified by x.

Syntax
:REF[x]:WFMLabel:XPOS <NR3>
:REF[x]:WFMLabel:XPOS?

Arguments
<NR3> is the desired X-position, in divisions from the left to right of the graticule. The range is 0 to 10.

Returns
The X-position of the specified reference waveform label in divisions of the graticule.

Examples
- :REF1:WFMLabel:XPOS 2 positions the Reference 1 waveform's label to 2 divisions from the left corner of graticule of the display.
- :REF1:WFMLabel:XPOS? Might return REF1:WFMLABEL:XPOS 2.0000000000 indicating that the Reference 1 waveform's label is positioned 2 divisions from the left corner of graticule of the display.

:REF[X]:WFMLabel:YPOS

Description
This command sets or queries the Y-position of the specified reference waveform label. The reference waveform is specified by x.

Syntax
:REF[x]:WFMLabel:YPOS <NR3>
:REF[x]:WFMLabel:YPOS?

Arguments
<NR3> is the desired Y-position, in divisions from zero line of the waveform. The range is -10 to +10.

Returns
The Y-position of the specified reference waveform label in divisions of the graticule.

Examples
- :REF1:WFMLabel:YPOS -3 positions the Reference 1 waveform's label to 3 divisions below the zero line of the waveform.

- :REF1:WFMLabel:YPOS? Might return REF1:WFMLABEL:YPOS 2.0000000000 indicating that the Reference 1 waveform's label is positioned 2 divisions above zero line of the waveform.

M[n]{A|B}:DESKew

Description
This command sets or queries the deskew time for the channel specified by channel M[n]{A|B}.

Syntax
:M[n]{A|B}:DESKew <NR3>
:M[n]{A|B}:DESKew?

Arguments
NR3 is the deskew time for this channel. The range is 0 to +100 ns with a resolution of 1 ps. Out of range values are clipped.

Returns
NR3 is the deskew value specified by channel M[n]{A|B}..

Examples
- :M1A:DESKEW 5.0E-9 sets the deskew time for Channel M1A to 5 ns.
- :M1B:DESKEW? might return M1B:DESKEW 10.00000000008E-5

M[n]{A|B}:EXTAtten:VALue

Description
This command sets or queries the value of the user-specified external attenuation factor. This external attenuation factor is meant to match the amount of attenuation applied externally, before the signal enters the specified input channel.

Syntax
:M[n]{A|B}:EXTAtten:VALue <NR3>
:M[n]{A|B}:EXTAtten:VALue?

Arguments
<NR3> is an external attenuation factor to apply to the acquired waveform. Value must be greater than zero.

Returns
An NR3 value representing the user-specified external attenuation factor.

Examples
- :M1B:EXTATTEN:VALUE 1.5 sets the external attenuation factor for channel B on module 1 to 1.500.
- :M2A:EXTATTEN:VALUE? might return 800.0000000000E-3, indicating the external attenuation factor for channel A on module 2 is 0.800.

M[n]{A|B}:EXTAtten:UNIts

Description
This command sets or queries the external attenuation unit to dB or Linear.

Syntax

:M[n]{A|B}:EXTAtten:UNIts { DB | LINear }
:M[n]{A|B}:EXTAtten:UNIts?

Arguments
DB sets external attenuation to dB.
LINear sets external attenuation to Linear.

Returns
DB means external attenuation is set to dB.
LINear means external attenuation is set to Linear.

Examples
- :M1B:EXTATTEN:UNIts LINear sets the external attenuation unit for channel B on module 1 to Linear.
- :M2A:EXTATTEN:UNIts? might return :DB, indicating the external attenuation units for channel A on module 2 are set to dB.

[n]{A|B}:OFFSet

Description

The user shall use this PI command to set or query the offset position of the specified live waveform. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
M[n]{A|B}:OFFSet <NR3>
M[n]{A|B}:OFFSet?

Arguments
<NR3> is the desired offset. The range is ±0.006W

Returns
The offset position of the specified waveform.

Examples
- M1A:OFFSET 1E-3 sets the module 1, channel A waveform offset to be 0.001W.
- M1A:OFFSET? might return 1.0000000000E-3.

DISPlay:M[n]{A|B}

Description
The user shall use this PI command to set or query whether the specified live waveform is displayed. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:DISPlay:M[n]{A|B} { ON | OFF | 0 | 1 }
:DISPlay:M[n]{A|B}?

Arguments
{ ON | OFF | 0 | 1 }: ON or 1 displays the specified live waveform. OFF or 0 turns off the display of the specified live waveform.

Returns
It returns the status of the waveform display.

Examples
- :DISPLAY:M1A ON displays M1A waveform .
- :DISPLAY:M1A? might return 0 to signify that the M1A waveform is not currently being displayed.

M[n]{A|B}:POSition

Description
The user shall use this PI command to set or query the vertical position of the specified live waveform.
The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
M[n]{A|B}:POSition <NR3>
M[n]{A|B}:POSition?

Arguments
<NR3> is the desired position, in divisions from the center graticule. The range is ±1000 divisions.

Returns
The vertical position of the specified waveform.

Examples
- M1A:POSITION 1.3 positions the module 1, channel A waveform 1.3 divisions above the center of the display.
- M1A:POSITION? Might return M1A:POSITION 2.0000000000, indicating that the M1A waveform is positioned 2 divisions above the center of the display.

M[n]{A|B}:UNITs? (Query Only)

Description
The user shall use this PI command to query the units associated with the live waveform specified.
The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
M[n]{A|B}:UNITs?

Returns
Return the vertical scale unit for the live waveform.

Examples
M1A:UNITS? Might return :M1A:UNITS “W”, indicating that the vertical scale units for the M1A waveform is set to “W”.

M[n]{A|B}:WFMLabel

Description
The user shall use this PI command to set or query the label associated with the waveform specified.

The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
M[n]{A|B}:WFMLabel <string>
M[n]{A|B}:WFMLabel?

Arguments
<string> takes a quoted string to set as the label for the waveform.

Returns
It returns the waveform label specified by the PI command.

Examples
- M1A:WFMLABEL "MY M1A WAVEFORM" sets the label for M1A to be "My M1A waveform", which is the label displayed with the waveform when it is displayed on screen.
- M1A:WFMLABEL? Might return M1A:WFMLABEL "MY M1A WAVEFORM", indicating that the label for the M1A waveform is set to "My M1A waveform".

M[n]{A|B}:SCAle

Description
The user shall use this PI command to set or query the vertical scale of the specified waveform.
The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
M[n]{A|B}:SCAle <NR3>
M[n]{A|B}:SCAle?

Arguments
<NR3> is the vertical reference scale in units per division.

Returns
The vertical scale of a live waveform.

Examples
- M1A:SCALE 1.0E-02 sets the module 1 channel A Waveform scale to 10 mV per division.
- M1A:SCALE? Might return M1A:SCALE 10.0000000000E-3, indication that the current volts per division setting of the module 1 channel A Waveform is 10 mV per division.

M[n]{A|B}:WFMLabel:XPOS

Description
This command sets or queries the X-position of the specified Live waveform label. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
M[n]{A|B}:WFMLabel:XPOS <NR3>
M[n]{A|B}:WFMLabel:XPOS?

Arguments
<NR3> is the X-position, in divisions from the left to right of the graticule. The range is 0 to 10.

Returns

The X-position of the specified Live waveform label in divisions of the graticule.

Examples
- M1A:WFMLabel:XPOS 3 positions the module 1 channel A waveform's label to 3 divisions from the left corner of graticule of the display.
- M1A:WFMLabel:XPOS? Might return M1A:WFMLABEL:XPOS 2.0000000000 indicating that the M1A waveform's label is positioned 2 divisions from the left corner of graticule of the display.

M[n]{A|B}:WFMLabel:YPOS

Description
This command sets or queries the Y-position of the specified Live waveform label. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
M[n]{A|B}:WFMLabel:YPOS <NR3>
M[n]{A|B}:WFMLabel:YPOS?

Arguments
<NR3> is the desired Y-position, in divisions from zero line of the waveform. The range is -10 to +10.

Returns
The Y-position of the specified Live waveform label in divisions of the graticule.

Examples
- M1A:WFMLabel:YPOS -3 positions the module 1 channel A waveform's label to 3 divisions below the zero line of the waveform.
- M1A:WFMLabel:YPOS? might return, M1A:WFMLABEL:YPOS 2.0000000000 indicating that the M1A waveform's label is positioned 2 divisions above the zero line of the waveform.

M[n]{A|B}:PERSistence

Description
This command sets or queries the persistence state for the specified channel.

Syntax
:M[n]{A|B}:PERSistence { ON | OFF | 1 | 0 }
:M[n]{A|B}:PERSistence?

Related Commands
:M[n]{A|B}:PERSistence:TYPe
:M[n]{A|B}:PERSistence:COUNt

Arguments
- ON or 1 enables persistence.
- OFF or 0 disables persistence.

Returns
Either 1 indicating persistence enabled or 0 indicating persistence disabled.

Examples
- :M1A:PERSISTENCE ON enables persistence for channel A on module 1.
- :M2B:PERSISTENCE? might return :M2B:PERSISTENCE 0 indicating that persistence is disabled for channel B on module 2.

M[n]{A|B}:PERSistence:TYPe

Description
This command sets or queries the persistence type for the specified channel.

Syntax
:M[n]{A|B}:PERSistence:TYPe { INFinite | VARiable }
:M[n]{A|B}:PERSistence:TYPe?

Related Commands
:M[n]{A|B}:PERSistence
:M[n]{A|B}:PERSistence:COUNt

Arguments
- INFinite indicates infinite persistence.
- VARiable indicates persistence using a configurable number of acquisitions.

Returns
Returns the persistence type for the specified channel.

Examples
- :M1B:PERSISTENCE:TYPE INFINITE sets the persistence type of channel B on module 1 to infinite persistence.
- :M2A:PERSISTENCE:TYPE? might return :M2A:PERSISTENCE:TYPE VARIABLE indicating that the persistence type of channel A on module 2 is set to variable persistence.

M[n]{A|B}:PERSistence:COUNt

Description
This command sets or queries the variable persistence count for the specified channel. Persistence count for a given channel can be set or queried at any time. However, it is only in effect when persistence is enabled for that channel, and that channel's persistence type is set to variable.

Syntax
:M[n]{A|B}:PERSistence:COUNt <NR1>
:M[n]{A|B}:PERSistence:COUNt?

Related Commands
M[n]{A|B}:PERSistence
M[n]{A|B}:PERSistence:TYPe

Arguments
<NR1> is the number of acquisitions to include in variable persistence.

Returns
The variable persistence count for the specified channel.

Examples
- :M1A:PERSISTENCE:COUNT 1200 sets the variable persistence count for channel A on module 1 to 1200.
- :M2A:PERSISTENCE:COUNT? might return :M2A:PERSISTENCE:COUNT 40 indicating that the variable persistence count for channel A on module 2 is set to 40.

:M[n]{A|B}:FILTer:TYPE

Description

This command sets or queries the filter type for the specified live channel. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:FILTer:TYPE { HW | BWE }
:M[n]{A|B}:FILTer:TYPE?

Arguments
- HW sets the specified channel's filter type to hardware filter.
- BWE sets the specified channel's filter type to BWE filter.

Returns
This query returns the filter type for the specified live channel.

Examples
- :M1A:FILTER:TYPE HW sets the filter type for channel M1A to Hardware Filter.
- :M2A:FILTER:TYPE? might return :M2A:FILTER:TYPE BWE indicating that the filter type for channel M2A is BWE Filter.

:M[n]{A|B}:HWFILTer:BW

Description
This command sets or queries the filter that is applied to the specified channel. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:HWFILTer:BW <NR3>
:M[n]{A|B}:HWFILTer:BW?

Related Commands
:M[n]{A|B}:HWFILTer:LIST?

Arguments
<NR3> is the filter bandwidth value in Hz, it should be selected from { 12.6e9 | 13.28125e9 | 26.5625e9 }

Returns
This query returns the filter bandwidth value in Hz.

Examples
:M2A:HWFILTER:BW? might return :M2A:HWFILTER:BW 12.6E+9, indicating that the hardware filter is 12.6GHz.

:M[n]{A|B}:HWFILTer:LIST? (Query Only)

Description
This query only command returns a list of the filters available for the specified channel. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:HWFILTer:LIST?

Related Commands
:M[n]{A|B}:HWFILTer:BW
:M[n]{A|B}:FILTer:TYPE

Returns
This query returns a list of the available filters for the specified channel.

Examples
:M1A:HWFILTER:LIST? might return
:M1A:HWFILTER:LIST 12.6000000000E+9,13.2812500000E+9,19.3350000000E+9, 21.0000000000E,22.5000000000E indicating the filters available for Module 1 Channel A.

:M[n]{A|B}:WLENgth

Description
This command sets or queries the wavelength value for the specified live waveform. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:WLENgth <NR1>
:M[n]{A|B}:WLENgth?

Related Commands
:M[n]{A|B}:WLENgth:LIST?

Arguments
NR1 specifies the wavelength value in nanometers.

Returns
The query form of this command returns the wavelength value for the specified channel in nanometers.

Examples
- :M1A:WLENGTH 1310 sets the Module 1 Channel A wavelength to 1310nm.
- :M1B:WLENGTH? might return :M1B:WLENGTH 1550, indicating that the wavelength for Module 1 Channel B is set to 1550nm.

:M[n]{A|B}:WLENgth:LIST? (Query Only)

Description
This query only command returns a list of the available wavelengths in nanometers for the specified live waveform. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module. The returned list contains the actual wavelengths available for the channel.

Syntax
:M[n]{A|B}:WLENgth:LIST?

Related Commands
:M[n]{A|B}:WLENgth

Returns
This query returns a list of available wavelengths in nanometers.

Examples

:M2A:WLENGTH:LIST? might return :M2A:WLENGTH:LIST 850,1310,1550, “User3”, “User15”, “User20”, indicating the available wavelengths for Module 2 Channel A.

M[n]{A|B}:AOP? (Query only)

Description
It returns measured Average Optical Power (AOP) for each optical channel in dBm or W. It is query only command.

Syntax
:M[n]{A|B}:AOP?

Related Commands
:M[n]{A|B}:AOP:UNIts {DBM | W}

Returns
The AOP value in dBm or W for the corresponding channel.

Examples
:M1A:AOP? might return the :M1A:AOP 2 indicating an Average Optical Power value of 2 dBm for channel A on module 1.

M[n]{A|B}:AOP:UNIts {DBM | W}

Description
It sets the Average Optical Power (AOP) unit to dBm or W.

Syntax
:M[n]{A|B}:AOP:UNIts {DBM | W}
:M[n]{A|B}:AOP:UNIts?

Related Commands
:M[n]{A|B}:AOP?

Arguments
DBM sets the AOP unit to dBm.
W sets the AOP unit to watts.

Returns
The AOP unit in dBm or W.

Examples
- :M[n]{A|B}:AOP:UNIts W sets the AOP unit to watts.
- :M[n]{A|B}:AOP:UNIts? might return M[n]{A|B}:AOP:UNIT W

:M[n]{A|B}:BWEFilter:BW

Description
This command sets or queries the BWE filter bandwidth that is applied to the specified channel. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:BWEFilter:BW <NR3>
:M[n]{A|B}:BWEFilter?

Related Commands
:M[n]{A|B}:BWEFilter:LIST?

Arguments
<NR3> is the BWE filter bandwidth value in Hz. It is selected from {11.2e9 | 13.28125e9 | 26.5625e9}. The list of available bandwidth selections is subject to change.

Returns
This query returns the BWE filter bandwidth value applied to the specified channel.

Examples
:M2A:BWEFilter:BW? might return :M2A:BWEFilter:BW 11.2000000000E+9, indicating that the 11.2 GHz BWE filter is enabled for Module 2 Channel A.

:M[n]{A|B}:BWEFilter:LIST?

This query only command returns a list of the BWE filters available for the specified channel. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:BWEFilter:LIST?

Related Commands
:M[n]{A|B}:BWEFilter:BW

Returns
This query returns a list of the available BWE filter bandwidth values for the specified channel.

Examples
:M1A:BWEFilter:LIST? might return
:M1A:FILTER:LIST 11.2000000000E+9,13.2812500000E+9,19.3350000000E+9,
21.0000000000E,22.5000000000E , indicating the BWE filters available for Module 1 Channel A.

:M[n]{A|B}:BWEFilter:STYPe

Description
This command sets or queries the BWE filter signal type that is applied to the specified channel. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:BWEFilter:STYPe {NRZ | PAM4}
:M[n]{A|B}:BWEFilter:STYPe?

Arguments
- "NRZ" sets the signal type to NRZ.
- "PAM4" sets the signal type to PAM4.

Returns

This query returns the signal type of the BWE filter applied to the specified channel.

Examples
:M2A:BWEFilter:STYPe? might return :M2A:BWEFilter:STYPe PAM4, indicating that the BWE filter for Module 2 Channel A is for PAM4 signaling.

:M[n]{A|B}:BWEFilter:ENABle?

Description
This command inquiries if the BWE filter is enabled or not. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:BWEFilter:ENABle?

Returns
This query returns 1 if the BWE filter is enabled, otherwise it returns zero.

Examples
:M2A:BWEFilter:ENABle? might return :M2A:BWEFilter:ENABle 1, indicating that the BWE filter is enabled.

:M[n]{A|B}:BWEFilter:ERROr?

Description
This command queries the error message occurred during BWE filter creation or applying process. Null is returned if no error occurs. The channel is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module.

Syntax
:M[n]{A|B}:BWEFilter:ERROr?

Returns
This query returns an error message if the BWE filter creation or apply failed otherwise it returns null.

Examples
:M2A:BWEFilter:ERROr? might return :M2A:BWEFilter:ERROr "BWE failed, ...", indicating that the BWE filter creation or apply failed.

M[n]{A|B}:WLENgth:USER[n]:CFACtor? (Query only)

Description
This command queries the correction factor of the selected user defined wavelength for the specified live waveform. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module. The user wavelength name is defined by USER[n].

Syntax
M[n]{A|B}:WLENgth:USER[n]:CFACtor?

Returns
<NR3> is the correction factor of the user defined wavelength in nm.

Examples

M1B:WLEN:USER3:CFACtor? might return 0.9, indicating that the correction factor for the user defined wavelength “User3” of the waveform on module 1 channel B is 0.9.

M[n]{A|B}:WLENgth:USER[n]:WLENGth? (Query only)

Description
This command queries the user defined wavelength for the specified live waveform. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module. The user wavelength name is defined by USER[n].

Syntax
M[n]{A|B}:WLENgth:USER[n]:WLENgth?

Returns
<NR3> is the wavelength value in nm.

Examples
M1B:WLEN:USER3:WLENgth? might return 1500.5, indicating that the user defined wavelength “User3” of the waveform on module 1 channel B is 1500.5 nm.

M[n]{A|B}:WLENgth:USER[n]:CLEAr (Command only)

Description
This command clears the correction factor and wavelength of the selected user defined wavelength. There is no query form. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module. The user wavelength name is defined by USER[n].

Syntax
M[n]{A|B}:WLENgth:USER[n]:CLEAr

Examples
M1B:WLEN:USER3:CLEAr will clear the user defined wavelength "User3" for M1B on module 1 channel B.

COMPensate:OPTGain:M[n]{A|B}:USER[n] (Command only)

Description
This command (no query form) compensates the gain of the optical channel by comparing the given input power to an average optical power measurement. The gain correction factor and the given input wavelength are automatically stored in the nonvolatile memory of the sampling module channel. The optical signal source must be connected and configured appropriately before sending this command. You should use a signal of known optical power at the wavelength for which you are compensating. The waveform is specified by M[n]{A|B}, where [n] is the module number and {A|B} is the channel name of the module. The user wavelength name is defined by USER[n].

Syntax
COMPensate:OPTGain:M[n]{A|B}:USER[n] <NR1>, <NR3>

Related Commands
M[n]{A|B}:WLEN:USER[n]:VALUe?
M[n]{A|B}:WLEN:USER[n]:CORRection?

Arguments

<NR1> is the wavelength in nm of the input optical signal.
<NR3> is the average optical power of the input optical signal in dBm. The input power value can be negative or positive. If the input value is smaller than the minimum value, it will be constrained to be the minimum value. The same applies to the maximum value.

Examples
COMPENSATE:OPTGain:M1A:USER3 1000,0.1 sets the wavelength to be 1000 nm and the power to be 0.1 dBm of channel M1A for user defined wavelength "User3".

Math Command Group

DISplay:MATH<x>

Description

This command will set or return whether or not the specified Math waveform is displayed. Define a Math waveform before turning the waveform on. <x> is the Math waveform.

Syntax
DISplay:MATH<x> { ON | OFF | 1 | 0 }
DISplay:MATH<x>?

Arguments
ON or 1 displays the specified waveform.
OFF or 0 turns off display of the specified waveform.
Any NR1 value other than 0 turns on display of the specified Math waveform.

Examples
DISPlay:MATH1 0 turns off the MATH1 display.

DISPlay:MATH1? Might return 0, indicating that MATH1 display is Off.

MATH:ADD (Command only)

Description
This command is used to define and add a Math waveform. The waveform is specified by [expression] in string format. It can be an arithmetic operation between the sources or a function.

Note: Define a reference waveform before turning the waveform on.

Syntax
MATH:ADD <expression> { }

Arguments
<expression> This can be a null string or an arithmetic operation between the sources or a function.

Examples
MATH:ADD "1A" defines and adds the source Math1 as '1A'.

MATH:DELete:ALL (Command only)

Description

This command deletes all the Math waveforms.

Syntax
MATH:DELete:ALL

Related Commands
MATH:ADD <expression>

Arguments
None.

Examples
MATH:DELete:ALL deletes all Math waveforms.

MATH:MATH<x>DELete (Command only)

Description
This command deletes the specified Math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:DELete

Related Commands
MATH:DELete:ALL

Arguments
None.

Examples
MATH:MATH1:DELete deletes the Math1 waveform.

MATH:MATH<x>:POSition

Description
This command sets or queries the vertical position of the specified Math waveform. <x> is the Math waveform. This command is equivalent to specifying the Vertical Position in the Math Configuration Menu (right-click property of the Math Badge in the Settings Bar at the bottom of the user interface).

Syntax
MATH:MATH<x>:POSition <NR3>
MATH:MATH<x>:POSition?

Arguments
<NR3> is the desired position, in divisions, from the center graticule.
Range: ±1000 divisions.

Returns
<NR3>, which is the vertical position if the specified Math waveform.

Examples
MATH:MATH2:POSITION 1.2E+000 positions the Math 2 waveform 1.2 divisions above the center of the display.

MATH:MATH1:POSITION? might return 1.2000000000, indicating that the Math1 waveform is positioned 1.2 divisions above the center of the display.

MATH:MATH<x>:SCALe

Description
This command sets or queries the vertical scale of the specified Math waveform. <x> is the Math waveform. This is equivalent to specifying the Vertical Scale in the Math Configuration Menu (right-click property of the Math Badge in the Settings Bar at the bottom of the user interface). Increasing the Scale causes the waveform to be displayed smaller. Decreasing the scale causes the waveform to be displayed larger. For Math waveforms, this setting controls the display only, graphically scaling these waveforms and having no effect on the acquisition hardware.

Syntax
MATH:MATH<x>:SCALe <NR3>
MATH:MATH<x>:SCALe?

Arguments
<NR3> is the vertical Math scale in units per division.

Returns
<NR3>, which is the vertical scale of the specified Math waveform.

Examples
MATH4:SCALE 1.0E-02 sets the Math 4 Waveform scale to 10 mV per division.
MATH:MATH1:SCALE? might return 10.0000000000E-3, indicating that the current volts per division setting of the Math1 waveform is 10 mV.

MATH:MATH<x>:STATus? (Query only)

Description
This command returns a Math Error/Warning as a string for the specified Math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:STATus?

Returns
A string of an informational message such as an error/warning.

Examples
MATH:MATH1:STATus? might return Error/Warning, which is the informational message for Math1.

MATH:MATH<x>:WFMLabel

Description
This command sets or queries the label associated with the specified Math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:WFMLabel <QString>

MATH:MATH<x>:WFMLabel?

Arguments
<QString> is a string that identifies the specified waveform.

Returns
Returns a string which is the label for the specified waveform.

Examples
MATH:MATH1:WFMLABEL “MY MATH1 WAVEFORM” sets the label for MATH1 to My MATH1 waveform. This is the label displayed on the screen with the waveform.
MATH:MATH1:WFMLABEL? might return MY MATH1 WAVEFORM, indicating that the label for the MATH1 waveform is MY MATH1 WAVEFORM.

MATH:MATH<x>:WFMLabel:XPOS

Description
This command sets or queries the X position of the specified Math waveform label. <x> is the Math waveform.

Syntax
MATH:MATH<x>:WFMLabel:XPOS <NR3>
MATH:MATH<x>:WFMLabel:XPOS?

Arguments
<NR3> is the desired X-position, in divisions, from the left to right of the graticule.
Range: 0 to 10.

Returns
The X-position of the specified Math waveform label in divisions of the graticule.

Examples
MATH:MATH1:WFMLabel:XPOS 2 positions the Math1 waveform label to 2 divisions from the left corner of graticule of the display.

MATH:MATH1:WFMLabel:XPOS? might return 2.0000000000, indicating that the Math1 waveform label is positioned 2 divisions from the left corner of graticule of the display.

MATH:MATH<x>:WFMLabel:YPOS

Description
This command sets or queries the Y position of the specified Math waveform label. <x> is the Math waveform.

Syntax
MATH:MATH<x>:WFMLabel:YPOS <NR3>
MATH:MATH<x>:WFMLabel:YPOS?

Arguments
<NR3> is the desired Y-position, in divisions, from the zero line of the waveform.
Range: -10 to +10.

Returns

The Y-position of the specified Math waveform label in divisions from the zero line of the waveform.

Examples
MATH:MATH1:WFMLabel:YPOS -3 positions the Math1 waveform label to 3 divisions below the zero line of the waveform.

MATH:MATH1:WFMLabel:YPOS? might return 2.0000000000, indicating that the Math1 waveform label is positioned 2 divisions above the zero line of the waveform.

MATH:MATH<x>:EXPRession

Description
This command sets or queries the Math expression associated with the specified Math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:EXPRession <string>
MATH:MATH<x>:EXPRession?

Arguments
<string> sets the expression for the Math waveform.

Returns
A string that is the expression for the Math waveform.

Examples
MATH:MATH1:EXPRession “1A” sets the expression for MATH1 to “1A”, which is the input source for Math1.

MATH:MATH1:EXPRession? might return 1A, indicating that the expression for the MATH1 waveform is set to 1A.

MATH:MATH<x>:CONFig:ATTRibutes (Query only)

Description
This command returns a list of available configuration attributes by name for the specified math source. <x> is the Math source.

Syntax
MATH:MATH<x>: CONFig:ATTRibutes?

Returns
A comma separated list of configuration attribute names. Returns an empty string if the specified math source has no configurations available.

Examples
MATH:MATH1:CONFig:ATTRibutes? might return "FoldAutoPosition,FoldHorizPosition", indicating that the math FOLD plugin configuration attributes are FoldAutoPosition, FoldHorizPosition.

MATH:MATH1:CONFig:ATTRibutes? might return "FFEAutoSet,ExtendedSearch", indicating that the math FFE plugin configuration attributes are FFEAutoSet, ExtendedSearch.

MATH:MATH<x>:CONFig

Description
This command sets or returns the value of configuration attribute for the given math source. The command sets the configuration value to the input value, if the input is valid (of correct type and/or range, where applicable). <x> is the Math source.

Use the query MATH:MATH<x>:CONfig? to find the Math source’s available configuration attributes.

Syntax
MATH:MATH<x>:CONFig <string_attribute>,{<string> | <NR3> | <Boolean>}
MATH:MATH<x>:CONFig? <string_attribute>

Arguments
<string_attribute> is the math configuration attribute name as a quoted string.
<string> if applicable, an attribute may be set to a string value.
<NR3> if applicable, an attribute may be set to a numeric value.
<Boolean> if applicable, an attribute may be set to a Boolean value

Returns
The value of the configuration attribute as an NR3, quoted string, or Boolean.

Examples
MATH:MATH1:CONFig "FFEAutoSet", OFF turns off the Autoset FFE of Math1.

MATH:MATH1:CONFig? "FFEAutoSet" might return “FFEAutoSet,1”, indicating that the Autoset FEE of Math1 is On.

MATH:MATH1:CONFig "FoldAutoPosition",OFF turns off the AutoPosition FOLD of Math1.

MATH:MATH1:CONFig? "FoldAutoPosition" might return "FoldAutoPosition,0”, indicating that the AutoPosition FOLD of Math1 is Off.

MATH:MATH<x>:PERSistence

Description
This command sets or returns the persistence state for the specified math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:PERSistence { ON | OFF | 1 | 0 }
MATH:MATH<x>:PERSistence?

Related Commands
MATH:MATH<x>:PERSistence:TYPe
MATH:MATH<x>:PERSistence:COUNt
MATH:MATH<x>:PERSistence:COLor

Arguments
ON or 1 enables persistence.
OFF or 0 disables persistence.

Returns
ON or 1 means persistence is enabled.

OFF or 0 means persistence is disabled.

Examples
MATH:MATH1:PERSISTENCE ON enables persistence for Math1.
MATH:MATH2:PERSISTENCE? might return 0, indicating that persistence is disabled for Math2.

MATH:MATH<x>:PERSistence:TYPe

Description
This command sets or returns the persistence type for the specified math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:PERSistence:TYPe { INFinite | VARiable }
MATH:MATH<x>:PERSistence:TYPe?

Related Commands
MATH:MATH<x>:PERSistence
MATH:MATH<x>:PERSistence:COUNt
MATH:MATH<x>:PERSistence:COLor

Arguments
INFinite sets the persistence type to Infinite.
VARiable sets the persistence type to a configurable number of acquisitions.

Returns
INFINITE
VARIABLE

Examples
MATH:MATH1:PERSistence:TYPE INFINITE sets the persistence type of Math1 to infinite persistence.

MATH:MATH2:PERSistence:TYPE? might return VARIABLE, indicating that the persistence type of Math2 is set to variable persistence.

MATH:MATH<x>:PERSistence:COUNt

Description
This command sets or returns the persistence count for the specified math waveform. Persistence count for a given Math waveform can only be set when the type is set to VARiable and persistence is enabled for that Math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:PERSistence:COUNt <NR1>
MATH:MATH<x>:PERSistence:COUNt?

Related Commands
MATH:MATH<x>:PERSistence
MATH:MATH<x>:PERSistence:TYPe
MATH:MATH<x>:PERSistence:COLor

Arguments

<NR1> is an integer which is the number of acquisitions to include in variable persistence.

Returns
An integer, which is the variable persistence count for the specified Math waveform.

Examples
MATH:MATH1:PERSistence:COUNt 1200 sets the persistence count of Math1 to 1200.

MATH:MATH2:PERSistence:COUNt? might return 40, indicating that the persistence count of Math2 is 40.

MATH:MATH<x>:PERSistence:COLor

Description
This command sets or returns the persistence color used for the specified math waveform. <x> is the Math waveform.

Syntax
MATH:MATH<x>:PERSistence:COLor { DSA300 | INVDSA300 | TEMPerature | INVTemperature | ENHTemperature | SPECTral | SPECial | NATive }
MATH:MATH<x>:PERSistence:COLor?

Related Commands
MATH:MATH<x>:PERSistence

Arguments
Available colors are: DSA300, INVDSA300, TEMPerature, INVTemperature, ENHTemperature, SPECTral, SPECial, NATive.

If the option selected is NATIVE, then that math source's color will be used for the persistence waveform trace.

Returns
The name of the persistence color used for the specified Math waveform.

Examples
MATH:MATH1:PERSistence:COLor SPECtral sets the persistence color of Math1 to Spectral.

MATH:MATH1:PERSistence:COLor? may return SPECTRAL, indicating that the persistence color of Math1 is Spectral.

Waveform Transfer Command Group

:CURVe

Description
The query form of this command transfers waveform data from TSOVu in binary format. The CURVe command transfers waveform data to TSOVu.

The query form of this command transfers waveform data from TSOVu in binary format. Use the DATa:SOUrce command to specify the location (source) of the waveform data that is transferred from TSOVu. The data encoding is specified by the DATa:ENCdg command, and the waveform format is specified by the DATa:FORMat command. In Vector format, the first and last data points that are transferred are specified by the DATa:STARt and DATa:STOP commands.

The CURVe command transfers waveform data to TSOVu. The data is stored in the reference memory location specified by DATa:DESTination, with first and last points specified by the DATa:STARt and DATa:STOP commands. If the destination reference is not empty, the reference slot will be cleared, and TSOVu will attempt to create a waveform using the provided data points.

Binary data can be represented by unsigned integer or floating-point values. The defined binary formats specify the order in which the bytes are transferred. The following are the available binary formats:

- FPBinary specifies floating-point data-point representation with the least significant byte transferred first.
- RPBinary specifies unsigned integer data-point representation with the least significant byte transferred first.

Syntax
:CURVe { <block> }
:CURVe?

Arguments
block is the waveform data in binary format. The waveform is formatted as: #<x><yyy><data> where <x> is the number of y bytes. For example, if <yyy> = 500, then <x> = 3. <yyy> is the number of bytes to transfer.

Returns
Waveform data configured through the DATa command set. At minimum, the returned data is from the source specified by DATa:SOUrce, in the format specified by DATa:FORMat, and in the encoding specified by DATa:ENCdg.

Examples
CURVE? might return :CURVE #3500<block>

:DATa:ENCdg

Description
This command sets or queries the format of the waveform data.

NOTE: Curve operations will honor this setting, regardless of the native encoding of a waveform in TSOVu. For example, vector waveforms in TSOVu are stored as floating-point values. If a user specifies :DATA:FORMAT VECTOR and :DATA:ENCDG RPBINARY, the following curve query would return vector data with each data point being an unsigned integer. This would almost certainly render the data useless.

Syntax
:DATa:ENCdg { FPBinary | RPBinary }
:DATa:ENCdg?

Related Commands
:CURVe

Arguments

- **FPBinary** specifies floating-point data-point representation with the least-significant byte transferred first.

- **RPBinary** specifies unsigned integer data-point representation with the least-significant byte transferred first.

Returns
The data-point representation of the waveform data output.

Examples
- :DATA:ENCDG FPBINARY sets the data-encoding format to floating point, with the least-significant byte transferred first.
- :DATA:ENCDG? might return :DATA:ENCDG RPBINARY, indicating that the format of the data is unsigned integer, with the least-significant bit transferred first.

:DATa:SOUrce

Description
This command sets or queries the location of the waveform data that is transferred from TSOVu by the CURVe? (query form).

Syntax
:DATa:SOUrce { M[n]{A|B} | REF[x] }
:DATa:SOUrce?

Related Commands
:CURVe?

Arguments
- **M[n]{A|B}** selects the specified channel acquisition waveform as the waveform source.
- **REF[x]** selects the specified Reference waveform as the waveform source.

Returns
The name of the source for the waveform data that is transferred using CURVe? (query form).

Examples
- :DATA:SOURCE REF2 specifies that the Reference 2 waveform will be transferred in the next CURVe? (query form).
- :DATA:SOURCE? might return :DATA:SOURCE REF3, indicating that Reference 3 waveform is the source for the waveform data that is transferred using CURVe? (query form).

:DATa:STARt

Description
Sets or queries the starting data point for waveform transfer. This command allows for the transfer of partial waveforms to and from TSOVu.
When executing a curve query, this setting is only taken into account if the data format is set to Vector (configured by the :DATa:FORMat command).

Syntax
:DATa:STARt <NR1>
:DATa:STARt?

Related Commands
:CURVe
:DATa:STOP

Arguments
<NR1> is the first data point that will be transferred. Data will be transferred from this point to DATa:STOP or the record length, whichever is less. If this value is greater than the record length, then no data will be transferred.

Returns
An integer value indicating the first waveform data point that will be transferred.

Examples
- :DATA:START 1 specifies that the waveform transfer will begin with the first data point.
- :DATA:START? might return :DATA:START 214, indicating that 214 is the first waveform data point that will be transferred.

:DATa:STOP

Description
Sets or queries the last data point that will be transferred when using the CURVe query and set commands. This allows the transfer of partial waveforms to and from TSOVu. When using the CURVe command, TSOVu will stop reading data when there is no more data to read.
When executing a curve query, this setting is only taken into account if the data format is set to Vector (configured by the :DATa:FORMat command).

Syntax
:DATa:STOP { END | <NR1> }
:DATa:STOP?

Related Commands
:CURVe
:DATa:STARt

Arguments
- **END** specifies that waveform data will be transferred from DATa:STARt to the last data point.
- **<NR1>** is the last data point that will be transferred. If fewer data points are provided than specified by the DATa:STOP command, then the resulting waveform is truncated.

Returns
Either END or an NR1 value indicating the last waveform data point that will be transferred.

Examples
- :DATA:STOP 4000 specifies that the waveform transfer will stop at data point 4000.
- :DATA:STOP? might return :DATA:STOP 500, indicating that 500 is the last data point that will be transferred.

:DATa:DESTination

Description
This command sets or queries the destination reference memory location for storing waveform data that is transferred into TSOVu by the CURVe set command.

Syntax
:DATa:DESTination REF[x]
:DATa:DESTination?

Related Commands

:CURVe

Arguments
REF[x] is the reference memory location where the waveform will be stored.

Returns
The reference memory location where the waveform will be stored.

Examples
- :DATA:DESTINATION REF3 stores the incoming waveform data in Reference 3.
- :DATA:DESTINATION? might return :DATA:DESTINATION REF8 as the reference memory location that is currently selected.

:DATa:WIDth? (Query Only)

Description
This command specifies the number of bytes per data point for waveform data transferred during a CURVe query or set command. This value is informed by the value of the :DATa:ENCdg setting.

Syntax
:DATa:WIDth?

Related Commands
:CURVe
:DATa:ENCdg

Returns
An NR1 value representing the number of bytes per data point for the waveform data transferred.

Examples
:DATA:WIDTH? might return :DATA:WIDTH 4, indicating that the width for each data point transferred will be 4 bytes.

:DATa:FORMat

Description
This command sets or queries the data format for a curve query. This setting only affects curve queries from TSOVu, not curve sets into TSOVu.

Syntax
:DATa:FORMat { VECTor | HITMap }
:DATa:FORMat?

Related Commands
:CURVe?
:DATa:ENCdg
:DATa:WIDth?

Arguments
- **VECTor** configures curve queries to return vector format data, which consists of sample values spaced equally in time.
- **HITMap** configures curve queries to return hitmap format data, which consists of a two-dimensional histogram of sample hits.

Returns

The data format for a curve query.

Examples
- :DATA:FORMAT VECTOR configures curve queries to return vector format data.
- :DATA:FORMAT? might return :DATA:FORMAT HITMAP, indicating that curve queries are configured to return hitmap format data.

:WFMInpre:XINcr

Description
This command sets or queries the time interval between samples of the incoming waveform.

Syntax
:WFMInpre:XINcr <NR3>
:WFMInpre:XINcr?

Arguments
<NR3> is the interval in seconds.

Returns
An NR3 value representing the time interval between samples of the waveform to be transferred.

Examples
- :WFMINPRE:XINCR 1.0E-9 sets the horizontal interval to 1 ns.
- :WFMINPRE:XINCR? might return :WFMINPRE:XINCR 5.0000000000E-9, indicating that the interval between samples is 5 ns.

:WFMInpre:XZEro

Description
This command sets or queries the time of first point of the waveform to be transferred. The time of first point is the time between the trigger point and the first point in the record.

Syntax
:WFMInpre:XZEro <NR3>
:WFMInpre:XZEro?

Arguments
<NR3> is the time of first point.

Returns
NR3 value representing the time of first point of the waveform to be transferred.

Examples
- :WFMINPRE:XZERO 20E-9 sets the time of first point to 20 ns for the incoming waveform.
- :WFMINPRE:XZERO? might return :WFMinpre:XZero 50.0000000000E-9, indicating that the time of first point in the incoming waveform record is 50 ns.

:WFMOutpre:HITMap:HBINs? (Query Only)

Description
This query-only command returns the number of bins in the horizontal axis of the last hitmap format curve query.

Syntax
:WFMOutpre:HITMap:HBINs?

Related Commands
:CURVe?
:WFMOutpre:HITMap:VBINs?
:DATa:FORMat

Returns
The number of bins in the horizontal axis of the last hitmap format curve query.

Examples
:WFMOUTPRE:HITMAP:HBINS? might return :WFMOUTPRE:HITMAP:HBINS 1000, indicating that the last hitmap format curve query returned data with 1000 bins in the horizontal axis.

:WFMOutpre:HITMap:VBINs? (Query Only)

Description
This query-only command returns the number of bins in the vertical axis of the last hitmap format curve query.

Syntax
:WFMOutpre:HITMap:VBINs?

Related Commands
:CURVe?
:WFMOutpre:HITMap:HBINs?
:DATa:FORMat

Returns
The number of bins in the vertical axis of the last hitmap format curve query.

Examples
:WFMOUTPRE:HITMAP:HBINS? might return :WFMOUTPRE:HITMAP:HBINS 750, indicating that the last hitmap format curve query returned data with 750 bins in the vertical axis.

Status and Events

Synchronization Methods

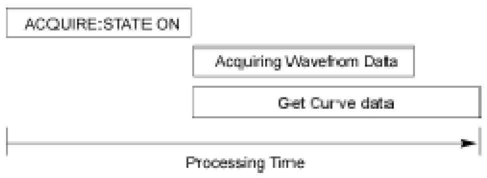
Although most GPIB commands are completed almost immediately after being received by the instrument, some commands start a process that requires more time. For example, once a single sequence acquisition command is executed, depending upon the applied signals and trigger settings, it may be a few seconds 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.

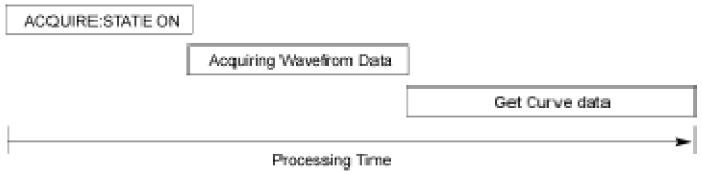
For example, synchronization may be used to ensure that the curve? query command returns waveform data that is consistent with the current instrument settings. You could use the following sequence to do this:

```
/** Sets up conditional acquisition **/
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 500
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER COUNT 1
ACQUIRE:STOPAFTER:MODE CONDITION
/** Sets up the data preamble **/
DATA:START 1
DATA:STOP 500
DATA:ENCDG RIBINARY
/** Clear data and acquire waveforms until conditional
stop occurs**/
ACQUIRE:DATA:CLEAR
ACQUIRE:STATE ON
/** Synchronize the operations by using *WAI, Busy?,
*OPC, or *OPC?/
<Synchronization command>
/** Get the curve data **/
CURVE?
```

The acquisition of the waveform requires extended processing time. It may not finish before the instrument executes the CURVe? query (see the following figure). This can result in incorrect curve values.



To ensure that the instrument completes waveform acquisition before attempting to execute the CURVe? query, you can synchronize the program. The figure below shows the desired processing sequence.



You can use four commands to synchronize the operation of the instrument with your application program: *WAI, BUSY?, *OPC, and *OPC?

***WAI** You can force commands to execute sequentially by using the *WAI command. This command forces completion of the previous commands before processing new ones.

The same command sequence using the *WAI command for synchronization looks like this:

```
/** Sets up conditional acquisition **/
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 500
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER COUNT 1
ACQUIRE:STOPAFTER:MODE CONDITION
/** Sets up the data preamble **/
DATA:START 1
DATA:STOP 500
DATA:ENCDG RIBINARY
/** Clear data and then acquire waveforms until
conditional stop occurs**/
ACQUIRE:DATA:CLEAR
ACQUIRE:STATE ON
/** Wait until the acquisition is complete before
querying the curve data**/
*WAI
/** Get the curve data **/
CURVE?
```

Although *WAI is one of the easiest ways to achieve synchronization, it is also the most costly. The processing time of the instrument is slowed since it is processing a single command at a time. This time could be spent doing other tasks.

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 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 timeout.

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:

```
/** Sets up conditional acquisition **/
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 500
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER COUNT 1
ACQUIRE:STOPAFTER:MODE CONDITION
/** Sets up the data preamble **/
DATA:START 1
DATA:STOP 500
DATA:ENCDG RIBINARY
/** Clear data and then acquire waveforms until
conditional stop occurs**/
ACQUIRE:DATA:CLEAR
ACQUIRE:STATE ON
/** Wait until the acquisition is complete before
querying the curve data**/
While BUSY? keep looping
/** Get the curve data **/
CURVE?
```

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:

```
/** Sets up conditional acquisition **/
```

```
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 500
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER COUNT 1
ACQUIRE:STOPAFTER:MODE CONDITION
/** Enable the status registers **/
DESE 1
*ESE 1
*SRE 0
/** Sets up the data preamble **/
DATA:START 1
DATA:STOP 500
DATA:ENCDG RIBINARY
/** Clear data and then acquire waveforms until
conditional stop occurs**/
ACQUIRE:DATA:CLEAR
ACQUIRE:STATE ON
/** Wait until the acquisition is complete before
querying the curve data**/
*OPC
While serial poll = 0, keep looping
/** Get the curve data **/
CURVE?
```

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:

```
/** Sets up conditional acquisition **/
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 500
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER:CONDITION ACQWFMS
ACQUIRE:STOPAFTER:COUNT 100
ACQUIRE:STOPAFTER:MODE CONDITION
/** Enable the status registers **/
DESE 1
*ESE 1
*SRE 32
```

```
/** Set up measurement parameters **/
MEASUREMENT:MEAS1:TYPE AMPLITUDE
MEASUREMENT:MEAS1:SOURCE CH1
/** Acquire waveforms until conditional stop occurs**/
ACQUIRE:DATA:CLEAR
ACQUIRE:STATE ON
/**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:MEAS1: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 is complete. A timeout 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:

```
/** Sets up conditional acquisition **/
ACQUIRE:STATE OFF
SELECT:CH1 ON
HORIZONTAL:RECORDLENGTH 500
ACQUIRE:MODE SAMPLE
ACQUIRE:STOPAFTER:COUNT 1
ACQUIRE:STOPAFTER:MODE CONDITION
/** Set up measurement parameters **/
MEASUREMENT:MEAS1:TYPE AMPLITUDE
EASUREMENT:MEAS1:SOURCE CH1
/** Clear data, and then acquire waveforms until
conditional stop occurs**/
ACQUIRE:DATA:CLEAR
ACQUIRE:STATE ON
/** Wait until the acquisition is complete
before querying the curve data**/
*OPC?
/** Wait for read from Output Queue **/
/** Get the curve data **/
CURVE?
```

This is the simplest approach. It requires no status handling or loops. However, you must set the controller timeout for a longer period of time than that used by the acquisition operation.

Messages

This section covers all the programming interface event 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 3-1: 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 3-2: 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

Table 3-2: Command Error Messages (CME Bit 5) (cont.)

Code	Message
112	Program mnemonic too long
113	Undefined header
114	Header suffix out of range
118	Query not allowed
120	Numeric data error
121	Invalid character in number
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
	Character data not allowed
150	String data error
151	Invalid string data
152	String data too long
158	String data not allowed
160	Block data error
161	Invalid block data
168	Block data not allowed
170	Expression error
171	Invalid expression
178	Expression data not allowed

Execution Error

The following table lists the execution errors that are detected during execution of a command.

Table 3-3: Execution Error Messages (EXE Bit 4)

Code	Message
200	Execution error
201	Invalid while in local
202	Settings lost due to RTL

Table 3-3: Execution Error Messages (EXE Bit 4) (cont.)

Code	Message
210	Trigger error
211	Trigger ignored
212	Arm ignored
213	Init ignored
214	Trigger deadlock
215	Arm deadlock
220	Parameter error
221	Settings conflict
222	Data out of range
223	Too much data
224	Illegal parameter value
225	Out of memory
226	List not same length
230	Data corrupt or stale
231	Data questionable
240	Hardware error
241	Hardware missing
242	Hardware configuration error
243	Hardware I/O device error
244	Invalid printer selected
250	Mass storage error
251	Missing mass storage
252	Missing media
253	Corrupt media
254	Media full
255	Directory full
256	File name not found
257	File name error
258	Media protected
260	Expression error
261	Math error in expression
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

Table 3-3: Execution Error Messages (EXE Bit 4) (cont.)

Code	Message
2205	Measurement error, Low amplitude
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 backwards Mid Ref Crossing
2212	Measurement error, No negative crossing
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
2224	Measurement error, WAIT calculating
2225	Measurement error, No waveform to measure
2226	Measurement error, Null Waveform
2227	Measurement error, Positive and Negative Clipping
2228	Measurement error, Positive Clipping
2229	Measurement error, Negative Clipping
2230	Measurement error, High Ref < Low Ref
2231	Measurement error, no statistics available
2235	Math error, Invalid math description
2236	Math error, Reference waveform is invalid
2237	Math error, Out of acquisition memory
2241	Waveform request is invalid
2243	This measurement cannot be performed on this type of waveform
2244	Source waveform is not active
2245	Saveref error, Selected channel is turned off
2246	Saveref error, Selected channel data invalid
2248	This ref cannot be activated
2249	Reference deletion error, Waveform in use

Table 3-3: Execution Error Messages (EXE Bit 4) (cont.)

Code	Message
2301	Cursor error, Off-screen
2303	Cursor error, Cursor source waveform is off
2304	Cursor error, Cursors are off
2321	Histogram warning, histogram turned off
2400	Not enough memory available
2401	This channel cannot be activated
2402	Math/Meas/Histo have circular definition
2410	Empty math string
2411	Syntax error in math string
2412	Semantic error in math string
2413	Math expression is too complex
2420	Histogram cannot be performed on this type of waveform
2425	Mask counting cannot be performed on this type of waveform
2430	WfmDB cannot be built on this type of waveform
2431	No Waveform Database resource available
2435	Selected Channel is not a TDR capable channel
2440	State Change – Reject Conditional Stop Action
2441	Selected Condition is Not Active
2450	TDR Invalid with FrameScan active
2451	Envelope Mode Invalid with FrameScan active
2455	RTL Hardware Warning
2456	RTL Software Error – severe problem
2457	RTL Software Warning
2458	Diagnostic Error
2459	Compensation Error
2500	Mask error – Mask Margin Boundary violated
2501	TDR coerced off – TDR operation not allowed with probe attached
2502	Waveform Database Source in use
2503	Clock Recovery not available
2504	Record length incompatible with scale
2505	Autoset Execute Warning – No waveforms enabled
2506	Error saving waveform to file
2507	FrameScan and Average Complete incompatible
2508	Selected filter is not available
2509	Gated Trigger is not available
2510	Warning – measurement requires waveform database, and none is available
2511	Error saving setup – invalid directory path or file

Table 3-3: Execution Error Messages (EXE Bit 4) (cont.)

Code	Message
2512	Could not close file
2513	Could not open file to read
2514	Can't write to file; media access violation
2515	Reference slot is out or range (1 – 8)
2516	Reference slot is active; cannot recall data into active slot
2517	Error reading waveform file
2518	Incompatible file version
2519	File path is not valid
2520	Waveform database is not properly released
2521	Reference data are not properly released
2522	Unable to restore setup
2523	Not enough mag points (<10) to save
2524	Directory does not exist; unable to create the directory
2525	Histogram contains invalid data
2526	The file can not be removed
2527	Source already active in another Database
2528	User-defined clock recovery not available
2529	TDR incompatible with Phase Correction Mode
2530	Phase Ref source not available
2531	TDR Step incompatible with phase Correction
2532	Phase Correction mode is off
2533	Clock Recovery source not available
2534	Clock Recovery rate not available
2535	Unit selection incompatible with Phase Ref mode
2536	Trigger Mode incompatible with Trigger Source
2537	Pattern Sync option not installed
2539	Pattern Sync Parameter not settable
2540	Autoset error, Failed to complete
2541	Autoset warning, Nothing to autoset
2542	Autoset error, Signal offset out of range
2543	Autoset error, Trigger not found
2544	Autoset error, Trigger Amplitude too small
2545	Autoset error, Signal Amplitude too small
2546	Autoset error, Signal Amplitude too large
2547	Autoset error, Signal period not found
2548	Autoset error, Eye not found
2549	Autoset error, Phase Ref Clock too slow

Table 3-3: Execution Error Messages (EXE Bit 4) (cont.)

Code	Message
2550	Application name error
2551	AutoSync error, No AutoSync on static wfm
2552	AutoSync error, No waveforms enabled
2553	AutoSync error, Mid Ref not found
2554	AutoSync error, Pattern Length not found
2555	AutoSync error, Data Rate not found
2556	AutoSync error, Trigger period not found
2557	AutoSync error, No AutoSync options selected
2558	Trigger source is not a clock source
2559	Autoset error, No TDR autoset on Ref wfm
2560	Autoset error, No TDR autoset on Math wfm
2561	Autoset error, Incident Edge not found
2562	Autoset error, Reflection not found
2563	Autoset error, Selected Channel is not a TDR capable channel
2570	Unused measurement slot not available
2571	Measurement failed
2572	Horizontal mask point violation
2573	Vertical mask point violation
2574	Mask counting not enabled
2575	Use Wfm Database not enabled
2576	Communication Standard not selected
2577	Mask must have three polygons
2578	Mask source and wfmDB vertical parameters don't match
2579	Phase Ref Triggered mode is not compatible with TDR trigger source
2580	Phase Ref Triggered mode is not compatible with Direct trigger source
2581	Phase Ref Triggered mode is not compatible with Free Run trigger source
2582	Phase Ref Triggered mode is not compatible with Other trigger mode
2583	Phase Ref Untriggered mode is not compatible with TDR trigger source
2584	Phase Ref Untriggered mode is not compatible with Direct trigger source
2585	Phase Ref Untriggered mode is not compatible with Clock trigger source
2586	Mask display not enabled
2587	Autoseek Mode is not HitRatio or MaskCount
2588	Selected Stop Action is not Available
2589	Autoseek could not find margin

Device Error

The following table lists the device errors that can occur during instrument operation. These errors may indicate that the instrument needs repair.

Table 3-4: 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
316	RTL Hardware Error, severe problem
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.

Table 3-5: 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)
420	Query UNTERMINATED (QYE bit 2 set)
430	Query DEADLOCKED (QYE bit 2 set)
440	Query UNTERMINATED after indefinite response (QYE bit 2 set)

Execution Warning

The following table lists warning messages that do not interrupt the flow of command execution. These notify you that you may get unexpected results.

Table 3-6: Execution Warning Messages (EXE Bit 4)

Code	Message
500	Execution warning
510	String data too long, truncated
525	Parameter underrange
526	Parameter overrange
527	Parameter rounded

Table 3-6: Execution Warning Messages (EXE Bit 4) (cont.)

Code	Message
528	Parameter out of range
530	Data stop > start. Values swapped internally
531	Data stop > record length, Curve truncated
532	Curve data too long, Curve truncated
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 minmax
546	Measurement warning, Need 3 edges
547	Measurement warning, Clipping positive/negative
548	Measurement warning, Clipping positive
549	Measurement warning, Clipping negative

Internal Warning

The following table shows internal errors that indicate an internal fault in the instrument.

Table 3-7: Internal Warning Messages

Code	Message
600	Internal warning

INDEX