DLM3022, DLM3032, DLM3052
Digital Oscilloscope
DLM3024, DLM3034, DLM3054
Mixed Signal Oscilloscope
Communication Interface

USER'S MANUAL

YOKOGAWA
Yokogawa Test & Measurement Corporation

IM DLM3054-17EN
1st Edition

Thank you for purchasing the DLM3022, DLM3032, or DLM3052 digital oscilloscope or the DLM3024, DLM3034, or DLM3054 mixed signal oscilloscope. This Communication Interface User's Manual explains the following interface features and commands.

- USB interface
- Ethernet interface
- GP-IB interface (option)

To ensure correct use, please read this manual thoroughly before operation.
After reading this manual, keep it in a safe place for quick reference in the event that a question arises.
The manuals for this instrument are listed on the next section. Please read all manuals.

Contact information of Yokogawa offices worldwide is provided on the following sheet.

Document No.	Description
PIM 113-01Z2	List of worldwide contacts

Notes

- The contents of this manual are subject to change without prior notice as a result of continuing improvements to the instrument's performance and functions. The figures given in this manual may differ from those that actually appear on your screen.
- Every effort has been made in the preparation of this manual to ensure the accuracy of its contents. However, should you have any questions or find any errors, please contact your nearest YOKOGAWA dealer.
- Copying or reproducing all or any part of the contents of this manual without the permission of YOKOGAWA is strictly prohibited.
- The TCP/IP software of this product and the documents concerning it have been developed/created by YOKOGAWA based on the BSD Networking Software, Release 1 that has been licensed from the Regents of the University of California.

Trademarks

- Microsoft, Internet Explorer, MS-DOS, Windows, Windows 7, Windows 8, Windows 8.1, and Windows 10 are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
- Adobe and Acrobat are either registered trademarks or trademarks of Adobe Systems Incorporated.
- DLM is a registered trademark of Yokogawa Electric Corporation.
- In this manual, the ® and TM symbols do not accompany their respective registered trademark or trademark names.
- Other company and product names are registered trademarks or trademarks of their respective holders.

1st Edition: November 2018 (YMI)

About the USB Interface and Ethernet Interface

To use the USB communication features, your PC must have the following:

- DLM Series Communication Library (TMCTL)
- USB device driver for connecting the DLM3000 series to the PC

To use the Ethernet communication features, your PC must have the following:

- DLM Series Communication Library (TMCTL)

To download the libraries and drivers listed above, go to the following website, and then browse to the download page.

https://tmi.yokogawa.com/library/

Sample Programs

To download sample programs, go to the following website, and then browse to the download page.

https://tmi.yokogawa.com/library/

Revisions

- 1st Edition: November 2018

Manuals

The following manuals, including this one, are provided as manuals for this instrument. Please read all manuals.

Manual Title	Manual No.	Description
DLM3022, DLM3032, DLM3052 Digital Oscilloscope DLM3024, DLM3034, DLM3054 Mixed Signal Oscilloscope Features Guide	IM DLM3054-01EN	The supplied CD contains the PDF file of this manual. This manual explains all the instrument's features other than the communication interface features.
DLM3022, DLM3032, DLM3052 Digital Oscilloscope DLM3024, DLM3034, DLM3054 Mixed Signal Oscilloscope User's Manual	IM DLM3054-02EN	The supplied CD contains the PDF file of this manual. The manual explains how to operate this instrument.
DLM3022, DLM3032, DLM3052 Digital Oscilloscope DLM3024, DLM3034, DLM3054 Mixed Signal Oscilloscope Getting Started Guide	IM DLM3054-03EN	Provided as a printed manual. This guide explains the handling precautions, common operations, troubleshooting measures, and specifications of this instrument.
DLM3022, DLM3032, DLM3052 Digital Oscilloscope DLM3024, DLM3034, DLM3054 Mixed Signal Oscilloscope Operation Guide	IM DLM3054-04EN	Provided as a printed manual. Explains the basic operations of this instrument. Operations are described in steps from "Preparation" to "Displaying Waveforms," "Measuring Waveforms," and "Saving Screen Captures."
DLM3022, DLM3032, DLM3052 Digital Oscilloscope DLM3024, DLM3034, DLM3054 Mixed Signal Oscilloscope Communication Interface User's Manual	IM DLM3054-17EN	This document. The supplied CD contains the PDF file of this manual. Explains the functions of the this instrument's communication interface, how to configure it, and how to control this instrument from a PC using the interface.
DLM3022, DLM3032, DLM3052 Digital Oscilloscope DLM3024, DLM3034, DLM3054 Mixed Signal Oscilloscope	IM DLM3054-92Z1	Document for China

The "EN" and "Z1" in the manual numbers are the language codes.

Manuals in the CD

The included CD (manual CD) contains the following English and Japanese manuals.

File Name	Manual No.	Description
Features Guide & Users Manual.pdf	IM DLM3054-01EN IM DLM3054-02EN	Features Guide and User's Manual
Communication Interface.pdf	IM DLM3054-17EN	Communication Interface User's Manual

HTML Help

A help document in HTML format stored in the instrument. The contents of this document are the same as the *Features Guide* (IM DLM3054-01EN). For the operating procedure, see section 3.10 in the *Getting Started Guide* (IM DLM3054-03EN).

How to Use This Manual

Structure of the Manual

This manual contains six chapters and an appendix.

Chapter 1 USB Interface

Describes the features and specifications of the USB interface.

Chapter 2 Ethernet Interface

Describes the features and specifications of the Ethernet interface and the specifications of the Socket interface.

Chapter 3 GP-IB Interface (Option)

Describes the features and specifications of the GP-IB interface.

Chapter 4 Programming Overview

Describes command syntax and other programming information.

Chapter 5 Commands

Describes every command individually.

Chapter 6 Status Reports

Describes the status byte, various registers, and queues.

Appendix

Provides reference material such as an ASCII character code table.

Index

Conventions Used in This Manual

Notes and Cautions

The notes and cautions in this manual are categorized using the following symbols.

WARNING Calls attention to actions or conditions that could cause serious or fatal injury to the user, and precautions that can be taken to prevent such occurrences.

CAUTION Calls attention to actions or conditions that could cause light injury to the user or damage to the instrument or user's data, and precautions that can be taken to prevent such occurrences.

French

AVERTISSEMENT Attire l'attention sur des gestes ou des conditions susceptibles de provoquer des blessures graves (voire mortelles), et sur les précautions de sécurité pouvant prévenir de tels accidents.

ATTENTION Attire l'attention sur des gestes ou des conditions susceptibles de provoquer des blessures légères ou d'endommager l'instrument ou les données de l'utilisateur, et sur les précautions de sécurité susceptibles de prévenir de tels accidents.

Note Calls attention to information that is important for proper operation of the instrument.

Character Notations

Hard Key Names and Soft Key Names in Bold Characters

Indicate panel keys that are used in the procedure and soft keys and menu items that appear on the screen.

Unit

k	Denotes 1000. Example: 100 kS/s (sample rate)
K	Denotes 1024. Example: 720 KB (file size)

Metasyntax

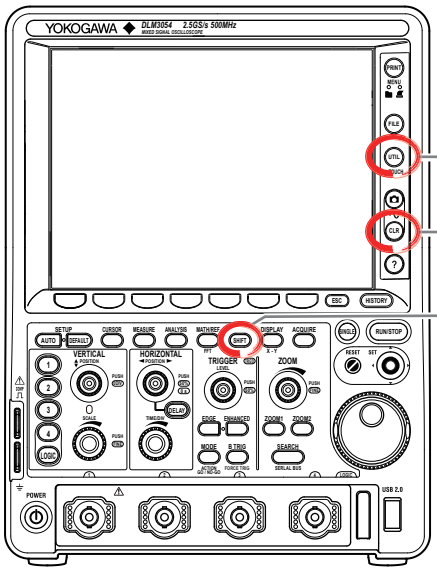
The following table contains the symbols that are used in the syntax discussed mainly in chapters 4 and 5. These symbols are referred to as BNF (Backus-Naur Form) symbols. For details on how to write data using these symbols, see pages 4-6 and 4-7.

Symbol	Description	Example	Example
<>	A defined value	CHANnel<x> <x> = 1 to 4	CHANNEL2
{ } |	Select an option in { } Exclusive OR	COUPling {AC|DC|DC50}	COUPLING AC
[]	Can be omitted	CURSor[:TY]:TYPE	CURSor:TYPE

Contents

1.1 Component Names and Functions

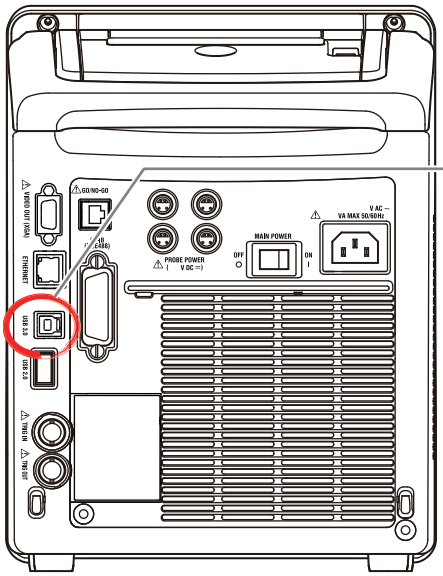
Front Panel



UTIL key
Press this key to select the USB interface.

SHIFT+CLR key
Press this key to switch to local mode. In local mode, remote mode (remote control using communication commands) is cleared, and key operation becomes possible.
However, key operation is invalid when Local Lockout (see page 1-3) is enabled by the controller.

Rear Panel



USB port for PCs
This port is for connecting the instrument to a controller (such as a PC) using a USB cable.

1.2 USB Interface Features and Specifications

USB Interface Features

Reception Features

Allow you to specify the same settings that you can using the front panel keys.
Receive output requests for measured and computed data, panel setting data, and error codes.

Transmission Features

The instrument can (1) transmit measured and computed data, (2) transmit panel setting data and the status byte, and (3) error codes when errors occur.

USB Interface Specifications

Electrical and mechanical specifications: USB 3.0/2.0
Connector: Type B connector (receptacle)
Ports: 1
Power supply: Self powered
PC system requirements: A PC running the English or Japanese version of Windows 7 (32 bit, 64 bit), Windows 8.1 (32 bit, 64 bit), or Windows 10 (32 bit, 64 bit) and a separate device driver

Data Transfer Rate

The following table contains approximations of how much time it takes for the instrument to transmit waveform data.
Model: DLM3054
Controller: PC (Intel (R) Core (TM) i5 CPU 650@3.20 GHz 3.33 GHz), USB2.0 (ICH7), OS (Windows 7) Enterprise 32 bit)
Programming language: Visual C++

Number of Data Points	Word Data
1250	4 ms
125000	9 ms
250000000	9505 ms

Switching between Remote and Local Modes

Switching from Local to Remote Mode

The instrument switches to Remote mode when it is in Local mode and it receives a `:COMMunicate:REMote ON` command from the PC.

- "REMOTE" appears at the top center of the screen once the instrument is in Remote mode.
- All keys except **SHIFT+CLR** key are disabled.
- The Local mode settings are retained even when the instrument switches to Remote mode.

Switching from Remote to Local Mode

When the instrument is in Remote mode and you press **SHIFT+CLR** key, the instrument switches to Local mode. However, this does not work if the instrument has received a `:COMMunicate:LOCKout ON` command from the PC. The instrument switches to Local mode when it receives a `:COMMunicate:REMote OFF` command from the PC, regardless of the local lockout state.

- The "REMOTE" indicator at the top center of the screen disappears once the instrument is in Local mode.
- All keys are enabled.
- The Remote mode settings are retained even when the instrument switches to Local mode.

Note

You cannot use the USB interface at the same time as other interfaces (GP-IB and Ethernet interfaces).

1.3 USB Interface Connection

Notes about Connections

- Be sure to insert the USB cable connector firmly into the USB port.
- If you are connecting multiple devices by using a USB hub, connect the instrument to the USB hub port that is closest to the port that the controller is connected to.
- Do not connect a USB cable to the GO/NO-GO output terminal. Doing so may damage the instrument.

1.4 Configuring the instrument USB Settings

This section explains the settings when controlling the instrument remotely through a USB interface.

Opening the Remote Control Menu

Press **UTIL**, and then press the **Remote Control** soft key to open the menu shown below.

Set the communication interface to USB.

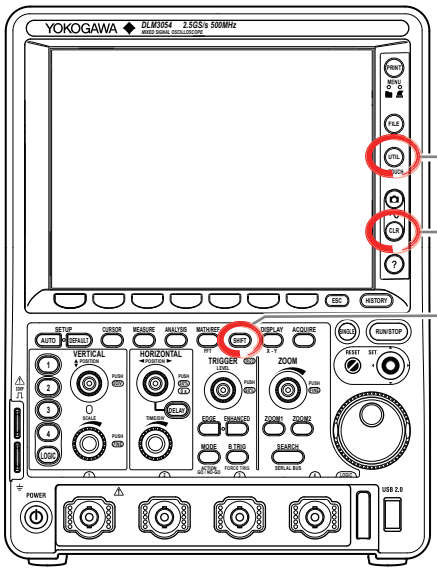
Set the mode to USB TMC.

Note

- If you send commands simultaneously from another communication interface that is not selected, the instrument will not execute the commands properly.
- To remotely control the instrument through a USB port using communication commands, set USB Function, shown in the figure above, to USB TMC, and then carry out the following steps.
  - To activate the USB Function settings, you need to restart the instrument. Turn off the instrument power switch, wait ten seconds or more, and then turn on the switch.
  - Install the YOKOGAWA USB driver on your PC. For information about how to obtain the YOKOGAWA USB driver, contact your nearest YOKOGAWA dealer. You can also access the YOKOGAWA USB driver download webpage and download the driver.
    https://tmi.yokogawa.com/library/
- Do not use USB drivers (or software) supplied by other companies.

2.1 Component Names and Functions

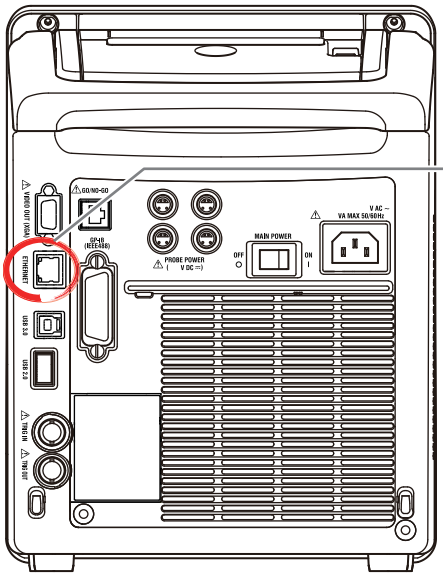
Front Panel



UTIL key
Press this key to select the Ethernet interface.

SHIFT+CLR key
Press this key to switch to local mode. In local mode, remote mode (remote control using communication commands) is cleared, and key operation becomes possible.
However, key operation is invalid when Local Lockout (see page 2-3) is enabled by the controller.

Rear Panel



Ethernet port
This port is for connecting the instrument to a controller (such as a PC) using an Ethernet cable.

2.2 Ethernet Interface Features and Specifications, Socket Interface Specifications

Ethernet Interface Features

Reception Features

Allow you to specify the same settings that you can using the front panel keys.
Receive output requests for measured and computed data, panel setting data, and error codes.

Transmission Features

The instrument can (1) transmit measured and computed data, (2) transmit panel setting data and the status byte, and (3) error codes when errors occur.

Ethernet Interface Specifications

Electrical and mechanical specifications: IEEE802.3
Simultaneous connections: 1
Protocol: VXI-11
PC system requirements: A PC running the English or Japanese version of Windows 7 (32 bit, 64 bit), Windows 8.1 (32 bit, 64 bit), or Windows 10 (32 bit, 64 bit)

Data Transfer Rate

The following table contains approximations of how much time it takes for the instrument to transmit waveform data.
Model: DLM3054
Controller: PC (Intel (R) Core (TM) i5 CPU 650@3.2 GHz 3.33 GHz), Ether (Intel (R) 82578DM Gigabit Network Connection), OS (Windows 7 Enterprise 32 bit)
Programming language: Visual C++

Number of Data Points	Word Data
1250	5 ms
125000	18 ms
250000000	18787 ms

Switching between Remote and Local Modes

Switching from Local to Remote Mode

The instrument switches to Remote mode when it is in Local mode and it receives a `:COMMunicate:REMote ON` command from the PC.

- "REMOTE" appears at the top center of the screen once the instrument is in Remote mode.
- All keys except **SHIFT+CLR** key are disabled.
- The Local mode settings are retained even when the instrument switches to Remote mode.

Switching from Remote to Local Mode

When the instrument is in Remote mode and you press **SHIFT+CLR** key, the instrument switches to Local mode. However, this does not work if the instrument has received a `:COMMunicate:LOCKout ON` command from the PC. The instrument switches to Local mode when it receives a `:COMMunicate:REMote OFF` command from the PC, regardless of the local lockout state.

- The "REMOTE" indicator at the top center of the screen disappears once the instrument is in Local mode.
- All keys are enabled.
- The Remote mode settings are retained even when the instrument switches to Local mode.

Note

You cannot use the Ethernet interface at the same time as other interfaces (GP-IB and USB interfaces).

Setting the Timeout Value

If the instrument is not accessed for a specified amount of time (set as a timeout value), the instrument closes the connection to the network. You can set the timeout value in the range of 1 s to 3600 s and Infinite. The default is Infinite. For instructions on how to set the timer value, see section 2.4, "Configuring the instrument Network Settings."

Socket Interface Specifications

Electrical and mechanical specifications: IEEE802.3
Simultaneous connections: 1
Protocol: None
Port number: 10002/tcp
Terminator: LF(0Ah)
Connection timeout: Infinite
PC system requirements: Windows 7 (32 bit, 64 bit), Windows 10 (32 bit, 64 bit)
Restriction: The number of bytes of command that can be sent at once is up to 4 Kbytes

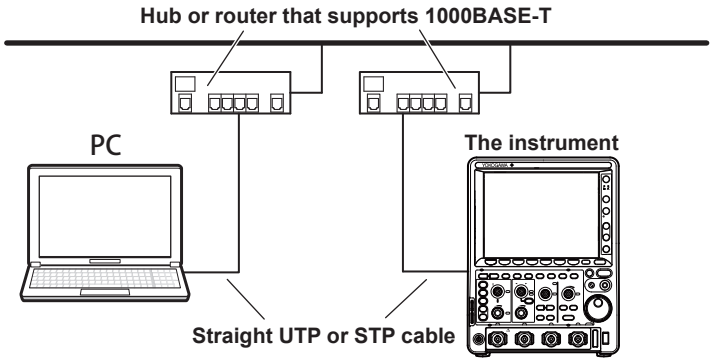
Socket Communication Sample Program

```
#include "stdafx.h"
#include <stdio.h>
#include <winsock2.h>
#include <ws2tcpip.h>

#pragma comment (lib,"Ws2_32.lib")

int _tmain(int argc, _TCHAR* argv[])
{
  /* Socket */
  int dstSocket;

  /* sockaddr_in structure */
  struct sockaddr_in dstAddr;

  /* Receive buffer */
  char buffer[200];

  /*************************************************************/
  /* Windows Socket API initialization */
  WSADATA data;
  WSAStartup(MAKEWORD(2,0), &data);

  /* Set the sockaddr_in structure */
  memset(&dstAddr, 0, sizeof(dstAddr));
  dstAddr.sin_port = htons(10002);
  dstAddr.sin_family = AF_INET;
  dstAddr.sin_addr.s_addr = inet_addr("192.168.0.1");

  /* Socket generation */
  dstSocket = socket(AF_INET, SOCK_STREAM, 0);

  /* Connection */
  connect(dstSocket, (struct sockaddr *) &dstAddr, sizeof(dstAddr));

  /* Packet transmission */
  send(dstSocket, "*IDN?\n", 6, 0);

  /* Packet reception */
  recv(dstSocket,buffer,200,0);

  /* Windows Socket API closure */
  closesocket(dstSocket);
  WSACleanup();

  return 0;
}
```

2.3 Ethernet Interface Connection

Connection Procedure

Connect a UTP (Unshielded Twisted-Pair) or STP (Shielded Twisted-Pair) cable that is connected to a hub or other network device to the Ethernet port on the instrument rear panel.



Notes about Connections

- To connect the instrument to a PC, be sure to use straight cables and to connect through a hub or router. Proper operation is not guaranteed for a one-to-one connection using a cross cable.
- Use a network cable that conforms to the transfer speed of your network.

Note

For details on how to connect the instrument to a network, see section 18.1, “Connecting the instrument to a Network” in the *instrument User’s Manual (IM DLM3054-02EN)*.

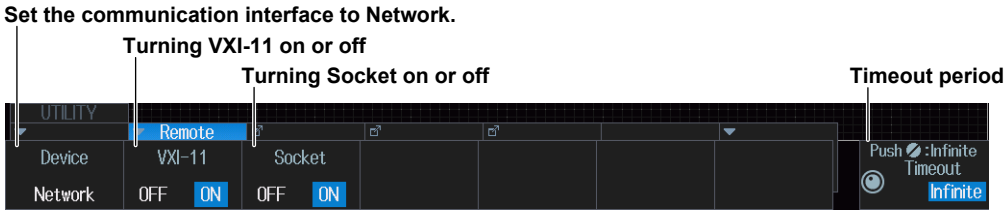
2.4 Configuring the instrument Network Settings

This section explains the settings listed below. You must configure these settings when controlling the instrument remotely through a ethernet interface.

•• Communication interface
- Turning the network connecting on or off (VXI-11 and Socket)
- Timeout period of the network

Opening the Remote Control Menu

Press **UTIL**, and then press the **Remote Control** soft key to open the menu shown below.



Note

- If you send commands simultaneously from another communication interface that is not selected, the instrument will not execute the commands properly.
- If VXI-11 or Socket is on, the Ethernet interface or the socket interface can be used. We recommend you turning them off if you are not using these features.

Configuring TCP/IP Settings

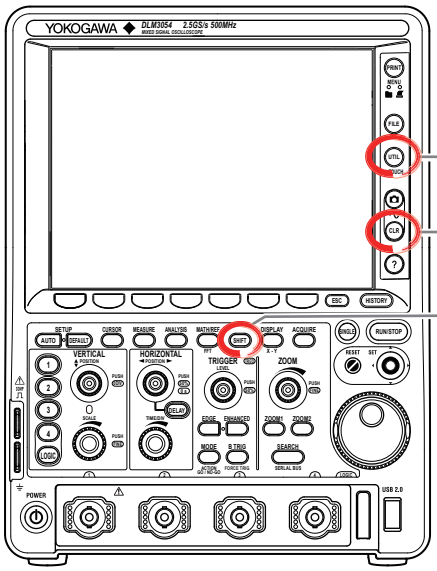
To use the Ethernet interface features, you must specify the following TCP/IP settings.

- IP address
- Subnet mask
- Default gateway

For instructions on how to specify these settings, see section 18.2, “Configuring TCP/IP Settings” in the *instrument User’s Manual (IM DLM3054-02EN)*.

3.1 Component Names and Functions

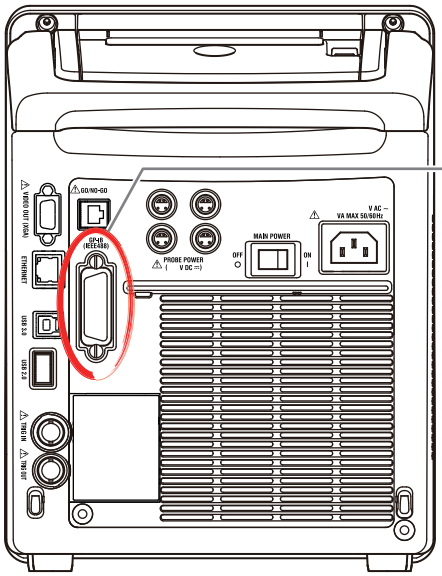
Front Panel



UTIL key
Press this key to select the GB-IB interface.

SHIFT+CLR key
Press this key to switch to local mode. In local mode, remote mode (remote control using communication commands) is cleared, and key operation becomes possible.
However, key operation is invalid when Local Lockout (see page 3-3) is enabled by the controller.

Left Side



GP-IB port
This port is for connecting the instrument to a controller (such as a PC) using a GP-IB cable.

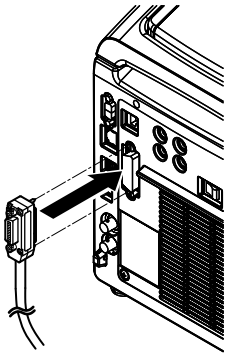
3.2 Connecting GP-IB Cables

GP-IB Cable

The instrument is equipped with a IEEE St'd 488-1978 24-pin GP-IB connector. Use GP-IB cables that complies with IEEE St'd 488-1978.

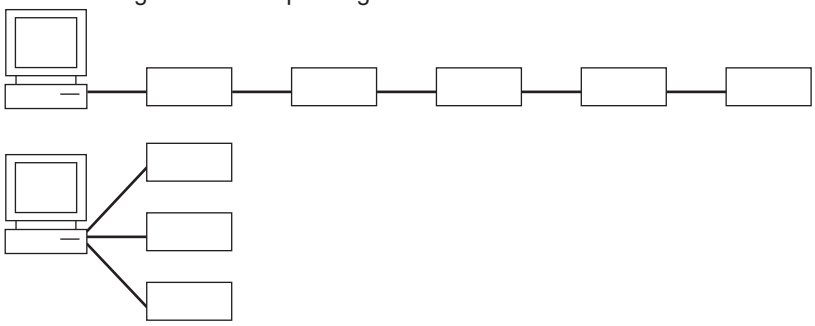
Connection Procedure

Connect a GP-IB cable as shown below.



Notes about Connections

- Securely fasten the GP-IB cable connector screws.
- On the PC end, use a GP-IB board (or card) made by National Instruments. For more details, see section 3.4.
- The instrument may not operate properly if the instrument is connected to the PC through converters (such as a GP-IB to USB converter). For more details, contact your nearest YOKOGAWA dealer.
- Several cables can be used to connect multiple devices. However, no more than 15 devices, including the controller, can be connected on a single bus.
- When connecting multiple devices, you must assign a unique address to each device.
- Use cables that are 2 m or shorter in length to connect devices.
- Keep the total length of the cables under 20 m.
- When devices are communicating, have at least two-thirds of the devices on the bus turned on.
- To connect multiple devices, use a daisy-chain or star configuration as shown below. You can also mix these configurations. Loop configuration is not allowed.



CAUTION

Be sure to turn off the PC and the instrument when connecting or removing communication cables. Otherwise, erroneous operation may result, or the internal circuitry may break.

French

ATTENTION

Veillez à mettre le PC et l'oscilloscope instrument hors tension lorsque vous branchez ou débranchez les câbles de communication, car cela risquerait de provoquer des dysfonctionnements ou des courts-circuits internes.

3.3 GP-IB Interface Features

GP-IB Interface Features

Listener Capabilities

- Allows you to specify the same instrument settings that you can using the front panel keys. You cannot turn the power on and off or change communication settings.
- Receive output requests for measured and computed data, panel setting data, and error codes.
- Receives status report commands and other commands.

Talker Capabilities

The instrument can (1) transmit measured and computed data, (2) transmit panel setting data and the status byte, and (3) error codes when errors occur.

Note

Talk-only, listen-only, and controller capabilities are not available on the instrument.

Switching between Remote and Local Modes

Switching from Local to Remote Mode

When the instrument is in Local mode and it receives a REN (Remote Enable) message from the PC, the instrument switches to Remote mode.

- "REMOTE" appears at the top center of the screen once the instrument is in Remote mode.
- All keys except **SHIFT+CLR** key are disabled.
- The settings in Local mode are retained even when the instrument switches to Remote mode.

Switching from Remote to Local Mode

When the instrument is in Remote mode and you press **SHIFT+CLR** key, the instrument switches to Local mode. This key is disabled when Local Lockout (see page 3-6 for details) has been activated by a controller.

- The "REMOTE" indicator at the top center of the screen disappears.
- All keys are enabled.
- The settings in Remote mode are retained even when the instrument switches to Local mode.

Note

You cannot use the GP-IB interface simultaneously with other interfaces (USB and Ethernet interfaces).

3.4 GP-IB Interface Specifications

GP-IB Interface Specifications

Electrical and mechanical specifications: IEEE St'd 488-1978

Functional specifications: See the table below.

Protocol: IEEE St'd 488.2-1992

Code: ISO (ASCII)

Mode: Addressable mode

Address setup: You can set the address to a number from 0 to 30 on the GP-IB setup screen that you can access from the UTILITY menu.

Clearing remote mode: You can clear Remote mode by pressing **SHIFT+CLR** key except when Local Lockout has been activated by the controller.

Functional Specifications

Function	Subset Name	Description
Source handshaking	SH1	Full source handshaking capability
Acceptor handshaking	AH1	Full acceptor handshaking capability
Talker	T6	Basic talker capability, serial polling, untalk on MLA (My Listen Address), and no talk-only capability
Listener	L4	Basic listener capability, unlisten on MTA (My Talk Address), and no listen-only capability
Service request	SR1	Full service request capability
Remote local	RL1	Full remote/local capability
Parallel polling	PP0	No parallel poll capability
Device clear	DC1	Full device clear capability
Device trigger	DT0	No device trigger capability
Controller	C0	No controller capability
Electric characteristics	E1	Open collector

Data Transfer Rate

The following table contains approximate response times for the instrument to transmit waveform data.

Model: DLM3054

Controller: PC (Intel (R) Core (TM) i5 CPU 650@3.20 GHz 3.33 GHz), GP-IB (NI PCIe-GP-IB), OS (Windows 7 Enterprise 32 bit)

Programming language: Visual C++

Number of Data Points	Word Data
1250	10 ms
125000	360 ms
250000000	723422 ms

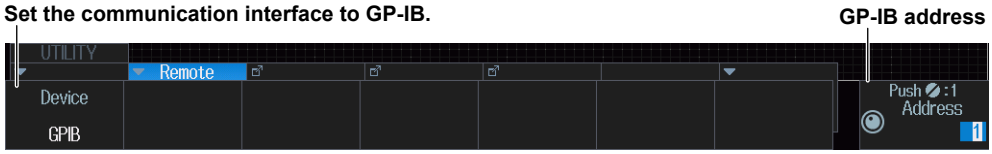
3.5 Configuring the Instrument GP-IB Settings

This section explains the settings listed below. You must configure these settings when controlling the instrument remotely through a GP-IB interface.

- Communication interface
- GP-IB address

Opening the Remote Control Menu

Press **UTIL**, and then press the **Remote Conrol** soft key to open the following menu.



Note

- If you send commands simultaneously from another communication interface that is not selected, the instrument will not execute the commands properly.
- When the controller is communicating with the instrument or with other devices through GP-IB, do not change the address.
- Each device that is connected by GP-IB has its own unique address in the GP-IB system. This address is used to distinguish one device from other devices. Therefore, you must assign a unique address to the instrument when connecting it to a PC or other device.

3.6 Responses to Interface Messages

Responses to Interface Messages

Responses to Uni-Line Messages

IFC (Interface Clear)

Clears the talker and listener functions. Stops data transmission if it is in progress.

REN (Remote Enable)

Switches between Remote and Local modes.

IDY (Identify) is not supported.

Responses to Multi-Line Messages (Address commands)

GTL (Go To Local)

Switches to Local mode.

SDC (Selected Device Clear)

- Clears the program message (command) being received and the output queue (see page 6-6 for details).
- Discards `*OPC` and `*OPC?` commands that are being executed.
- Immediately aborts `*WAI` and `COMMunicate:WAIT`.

PPC (Parallel Poll Configure), GET (Group Execute Trigger), and TCT (Take Control) are not supported.

Responses to Multi-Line Messages (Universal commands)

LLO (Local Lockout)

Disables the **SHIFT+CLR** key on the front panel to prohibit switching to the local mode.

DCL (Device Clear)

Performs the same operation as SDC.

SPE (Serial Poll Enable)

Sets the talker function on all devices on the bus to serial poll mode. The controller will poll each device in order.

SPD (Serial Poll Disable)

Clears the talker function's serial poll mode on all devices on the bus.

PPU (Parallel Poll Unconfigure) is not supported.

What Are Interface Messages?

Interface messages are commands that a controller transmits. They are also referred to as interface commands or bus commands. They are classified as follows:

Uni-line Messages

Uni-line messages are sent over a single control line. The following three messages are available.

- IFC (Interface Clear)
- REN (Remote Enable)
- IDY (Identify)

Multi-line Messages

Multi-line messages are sent over eight data lines. The messages are grouped as follows:

Address Commands

Address commands are valid when the instrument is designated as a listener or a talker. The following five commands are available.

Commands available to a device designated as a listener

- GTL (Go To Local)
- SDC (Selected Device Clear)
- PPC (Parallel Poll Configure)
- GET (Group Execute Trigger)

Commands available to a device designated as a talker

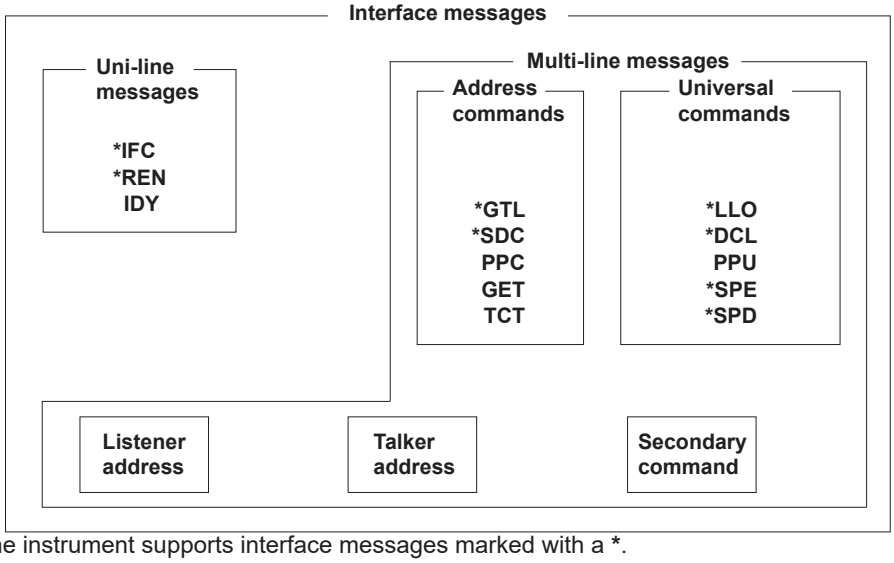
- TCT (Take Control)

Universal Commands

Universal commands are available to all devices regardless of their listener or talker designation. The following five commands are available.

- LLO (Local Lockout)
- DCL (Device Clear)
- PPU (Parallel Poll Unconfigure)
- SPE (Serial Poll Enable)
- SPD (Serial Poll Disable)

There are other interface messages: listener-address, talk-address, and secondary commands.



The instrument supports interface messages marked with a *.

Note

Difference between SDC and DCL

In multi-line messages, SDC messages are those that require talker or listener designation and DCL messages are those that do not require the designation. Therefore, the SDC command affects a specific device while the DCL command affects all devices on the bus.

4.1 Messages

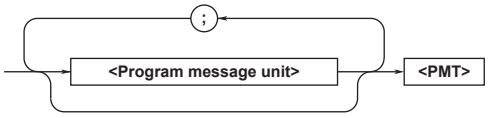
Messages

Messages are used to exchange information between the controller and the instrument. Messages that are sent from the controller to the instrument are called program messages, and messages that are sent from the instrument back to the controller are called response messages.

If a program message contains a command that requests a response (query), the instrument returns a response message upon receiving the program message. The instrument returns a single response message in response to a single program message.

Program Message

The program message syntax is as follows:



<Program Message Unit>

A program message consists of one or more program message units. Each unit corresponds to one command. The instrument executes the commands in the order that they are received.

Separate each program message unit with a semicolon.

For details on the program message syntax, see the next section.

Example

```
:ACQuire:MODE NORMal;RESolution 1<PMT>
```

Unit Unit

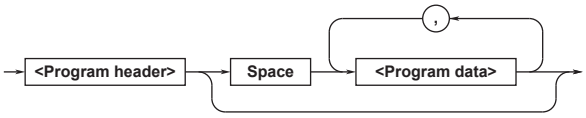
<PMT>

<PMT> is a program message terminator. The following three terminators are available.

NL (new line):	Same as LF (line feed). ASCII code “0AH”
^EOM:	The END message as defined by USBTMC (The data byte that is sent with the END message is the last data byte of the program message.)
NL^EOM:	NL with an END message attached. (NL is not included in the program message.)

Program Message Unit Syntax

The program message unit syntax is as follows:



<Program Header>

The program header indicates the command type. For details, see page 4-3.

<Program Data>

Attach program data if there are conditions that are required to execute a command. Separate the program data from the header with a space (ASCII code 20H).

If there are multiple data values, separate each data value with a comma.

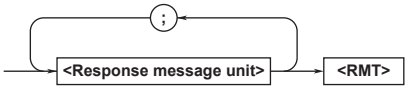
For details, see page 4-5.

Example `:ACQuire:MODE NORMal<PMT>`

Header Data

Response Message

The response message syntax is as follows:



<Response Message Unit>

A response message consists of one or more response message units; each response message unit corresponds to one response.

Separate each response message unit with a semicolon.

For details on the response message syntax, see the next page.

Example

```
:ACQUIRE:MODE NORMAL;RESolution 1<RMT>
```

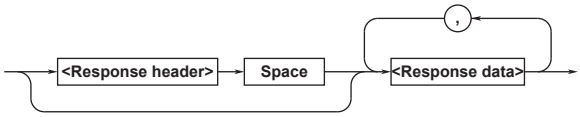
Unit Unit

<RMT>

RMT stands for “response message terminator.” The response message terminator is NL^EOM.

Response Message Unit Syntax

The response message unit syntax is as follows:



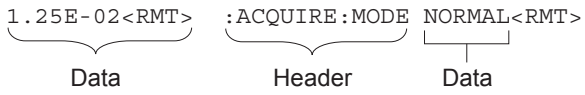
<Response Header>

A response header sometimes precedes the response data. A space separates the data from the header. For details, see page 4-5.

<Response Data>

Response data contains the content of the response. If there are multiple data values, each data value is separated by a comma. For details, see page 4-5.

Example



If there are multiple queries in a program message, responses are returned in the same order that the queries were received in. The instrument returns a single response message unit to most queries, but there are queries that the instrument returns multiple units to. The first response message unit always corresponds to the first query, but the n^{th} response unit may not necessarily correspond to the n^{th} query. If you want to make sure that every response is retrieved, divide the program messages into individual messages.

Precautions to Be Taken when Exchanging Messages

- If the controller sends a program message that does not contain a query, the controller can send the next program message at any time.
- If the controller sends a program message that contains a query, the controller must finish receiving the response message before it can send the next program message. If the controller sends the next program message before receiving the response message in its entirety, an error will occur. A response message that is not received in its entirety will be discarded.
- If the controller tries to receive a response message when there is none, an error will occur. If the controller tries to receive a response message before the transmission of the program message is complete, an error will occur.
- If the controller sends a program message containing multiple message units, but the message contains incomplete units, the instrument will try to execute the ones that are believed to be complete. However, these attempts may not always be successful. In addition, if such a message contains queries, the instrument may not necessary return responses.

Deadlock

The instrument can store at least 1024 bytes of messages in its transmit and receive buffers (the number of available bytes varies depending on the operating conditions). If both the transmit and receive buffers become full at the same time, the instrument will no longer be able to operate. This condition is called a deadlock. If this happens, you can resume operation after you have discarded response messages. Deadlock will not occur if the program message (including the <PMT>) is kept below 1024 bytes. Program messages that do not contain queries never cause deadlocks.

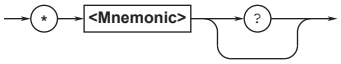
4.2 Commands

Command

There are three types of commands (program headers) that a controller may send to the instrument. The commands differ in their program header formats.

Common Command Header

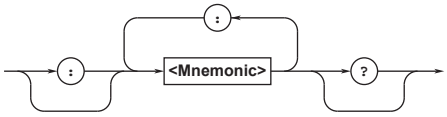
Commands that are defined in USBTMC-USB488 are called common commands. The header format of a common command is shown below. Be sure to include an asterisk (*) at the beginning of a common command.



Common command example *CLS

Compound Header

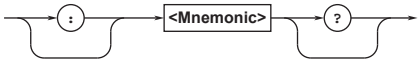
Other commands that are specific to the instrument are classified and arranged in a hierarchy according to their functions. The compound header syntax is shown below. Be sure to use a colon to specify a lower hierarchical level.



Compound header example :ACQuire:MODE

Simple Header

These commands are functionally independent and are not contained within a hierarchy. The format of a simple header is shown below.



Simple header example :STARt

Note

A <mnemonic> is an alphanumeric character string.

When Concatenating Commands

- **Command Groups**

  A command group is a group of commands that have common compound headers arranged in a hierarchy. A command group may contain sub-groups.

  Example Group of commands related to acquisition
  ```
  :ACQuire:AVERage:COUNt
  :ACQuire:MODE
  :ACQuire:RLENgth
  ```

- **When Concatenating Commands of the Same Group**

  The instrument stores the hierarchical level of the command that is currently being executed and processes the next command on the assumption that it belongs to the same level. Therefore, the common header section can be omitted for commands that belong to the same group.

  Example
  ```
  :ACQuire:MODE NORMal;
  SAMPling REAL<PMT>
  ```

- **When Concatenating Commands of Different Groups**

  If the subsequent command does not belong to the same group, place a colon in front of the header (cannot be omitted).

  Example
  ```
  :ACQuire:MODE NORMal;:DISPlay:
  FORMat SINGle<PMT>
  ```

- **When Concatenating Simple Headers**

  If a simple header follows another command, place a colon in front of the simple header (cannot be omitted).

  Example
  ```
  :ACQuire:MODE NORMal;:
  STARt<PMT>
  ```

- **When Concatenating Common Commands**

  Common commands that are defined in the USBTMC-USB488 are independent of hierarchy. There is no need to use a colon.

  Example
  ```
  :ACQuire:MODE NORMal;*CLS;
  SAMPling REAL<PMT>
  ```

- **When Separating Commands with <PMT>**
  If you separate two commands with a terminator, two program messages will be sent. Therefore, the common header must be specified for each command even if commands belonging to the same command group are being concatenated.
  Example `:ACQuire:MODE NORMal<PMT>:ACQuire:SAMPling REAL<PMT>`

Upper-Level Query

An upper-level query is a query that is made by appending a question mark to a command higher in the group. The controller can receive all of the settings in a group collectively by executing a highest-level query. Some upper-level queries of a group, which may be comprised of more than three hierarchical levels, can cause the instrument to transmit all the lower level settings.

Example

```
:ACQUIRE? -> :ACQUIRE:AVERAGE:COUNT 2;
ACQUIRE:COUNT INFINITY;MODE NORMAL;
RESOLUTION 0;RLENGTH 125000;
SAMPLING INTERPOLATE
```

The response to an upper-level query can be sent back to the instrument as a program message. This enables the settings that were present when the upper-level query was made to be reproduced later on. However, some upper-level queries do not return setup data that is not currently in use. Exercise caution because not all of a group's information is necessarily returned in a response.

Header Interpretation Rules

The instrument interprets the header that it receives according to the rules below.

- Mnemonics are not case sensitive.
  Example `CURSor` can be written as `cursor` or `Cursor`.
- The lower-case characters can be omitted.
  Example `CURSor` can be written as `CURSO` or `CURS`.
- The question mark at the end of a header indicates that it is a query. You cannot omit the question mark.
  Example: The shortest abbreviation for `CURSor?` is `CURS?`.
- If the <x> (value) at the end of a mnemonic is omitted, it is interpreted as a 1.
  Example: If you write `CHAN` for `CHANnel<x>`, `CHANnel1` is specified.
- Parts of commands and parameters enclosed in square brackets ([ ]) can be omitted.
  Example: `TRIGger[:ATRigger]:SIMPle:LEVel` can be written as `TRIG:SIMP:LEV`.
  However, the last section enclosed in brackets cannot be omitted in an upper-level query.
  Example: `TRIGger?` and `TRIGger:ATRigger?` are different queries.

4.3 Responses

Response

When the controller sends a query with a question mark, the instrument returns a response message to the query. The instrument returns response messages in one of the following two forms.

- **Response Consisting of a Header and Data**
  Responses that can be used as program messages without any changes are returned with command headers attached.
  Example `:ACQUire:MODE?<PMT>`
  `-> :ACQUire:MODE NORMAL<RMT>`

- **Response Only Consisting of Data**
  Responses that cannot be used as program messages unless changes are made (query-only commands) are returned without headers. However, there are query-only commands whose responses the instrument will attach headers to.
  Example `:MEASure:CHANnel1:PTOPeak:`
  `VALue?<PMT> -> 10.0E+00<RMT>`

If You Want the Instrument to Return Responses without Headers

You can configure the instrument so that even responses that have both headers and data are returned without headers. Use the `COMMunicate:HEADer` command for this purpose.

Abbreviated Form

The instrument normally returns response headers with the lower-case section removed. You can configure the instrument so that full headers are returned. Use the `COMMunicate:VERBose` command for this purpose. The sections enclosed in braces ([ ]) are also omitted in the abbreviated form.

4.4 Data

Data

Data contains conditions and values that are written after the header. A space separates the data from the header. Data is grouped as follows:

Data	Description
<Decimal>	A value expressed in decimal notation (Example: Probe attenuation for CH1 `-> CHANnel1:PROBe 100`)
<Voltage><Time> <Frequency> <Current>	A physical value (Example: Time-axis range `-> TIMebase:TDIV 1US`)
<Register>	A register value expressed as binary, octal, decimal or hexadecimal (Example: Extended event register value `-> STATUS:EESE #HFE`)
<Character data>	Predefined character string (mnemonic). Select from the available strings in braces. (Example: Select the input coupling of CH1 `-> CHANnel1:COUPling{AC|DC|DC50}`)
<Boolean>	Indicates ON and OFF. Specify ON, OFF, or a value (Example: Turn on the CH1 display `-> CHANnel1:DISPlay ON`)
<String data>	User-defined string (Example: Comment attached to screen data output `-> FILE:SAVE:COMMent`)
<Filename>	Indicates a file name. (Example: Save file name `-> FILE:SAVE:WAVeform:NAME "CASE1"`)
<Block data>	Data that contains 8-bit values (Example: Response to acquired waveform data `-> #800000010ABCDEFGHIJ`)

<Decimal>

<Decimal> indicates a value expressed as a decimal number, as shown in the table below. Decimal values are written in the NR form as specified in ANSI X3.42-1975.

Symbol	Description	Example		
<NR1>	Integer	`125`	`-1`	`+1000`
<NR2>	Fixed point number	`125.0`	`-.90`	`+001.`
<NR3>	Floating-point number	`125.0E+0`	`-9E-1`	`+.1E4`
<NRf>	Any form from <NR1> to <NR3>			

- The instrument can receive decimal values that are sent from the controller in any form, from <NR1> to <NR3>. This is expressed as <NRf>.
- The instrument returns a response to the controller in one of the forms from <NR1> to <NR3> depending on the query. The same form is used regardless of the size of the value.
- For the <NR3> form, the plus sign after the “E” can be omitted. You cannot omit the minus sign.
- If a value outside the setting range is entered, the value is adjusted to the closest value within the range.
- If a value has more significant digits than are available, the value will be rounded.

<Voltage>, <Time>, <Frequency>, <Current>

<Voltage>, <Time>, <Frequency>, and <Current> indicate decimal values that have physical significance. A <Multiplier> or <Unit> can be attached to the <NRf> form that was described earlier. The following types of expressions are possible.

Format	Example
<NRf><Multiplier><Unit>	`5MV`
<NRf><Unit>	`5E-3V`
<NRf>	`5E-3`

<Multiplier>

<Multipliers> that you can use are indicated in the following table.

Symbol	Word	Multiplier
EX	Exa	10^{18}
PE	Peta	10^{15}
T	Tera	10^{12}
G	Giga	10^{9}
MA	Mega	10^{6}
K	Kilo	10^{3}
M	Milli	10^{-3}
U	Micro	10^{-6}
N	Nano	10^{-9}
P	Pico	10^{-12}
F	Femto	10^{-15}
A	Atto	10^{-18}

<Unit>

<Units> that you can use are indicated in the following table.

Symbol	Word	Description
V	Volt	Voltage
S	Second	Time
HZ	Hertz	Frequency
MHZ	Megahertz	Frequency
A	Ampere	Current

- <Multiplier> and <Unit> are not case sensitive.
- “U” is used to indicate micro (“μ”).
- “MA” is used for Mega to distinguish it from Milli. Megahertz, which is expressed as “MHZ,” is an exception. Therefore, “M (Milli)” cannot be used for frequencies.
- If both <Multiplier> and <Unit> are omitted, the default unit is used.
- Response messages are always expressed in the <NR3> form. Response messages are returned using the default unit without the <Multiplier> or <Unit>.

<Register>

<Register> is an integer that can be expressed in decimal, hexadecimal, octal, or binary notation. It is used when each bit of the value has a particular meaning. The following types of expressions are possible.

Form	Example
<NRf>	1
#H<Hexadecimal value made up of the digits 0 to 9 and A to F>	#H0F
#Q<Octal value made up of the digits 0 to 7>	#Q777
#B<Binary value made up of the digits 0 and 1>	#B001100

- <Register> is not case sensitive.
- Response messages are always expressed in the <NR1> form.

<Character Data>

<Character data> is a predefined character string (mnemonics). It is mainly used to indicate that an option listed as a character string in braces must be selected and entered. The data interpretation rules are the same as those described in “Header Interpretation Rules” on page 4-4.

Form	Example
{AC|DC|DC50}	AC

- As with the header, the COMMunicate:VERBose command can be used to select whether to return the response in the full form or in the abbreviated form.
- The COMMunicate:HEADer setting does not affect <character data>.

<Boolean>

<Boolean> is data that indicates ON or OFF. The following types of expressions are possible.

Form	Example
{ON|OFF|<NRf>}	ON OFF 1 0

- When <Boolean> is expressed in the <NRf> form, “OFF” is selected if the rounded integer value is 0, and ON is selected for all other cases.
- A response message is always returned with a 1 if the value is ON and with a 0 if the value is OFF.

<String Data>

<String data> is not a predefined character string like <character data>. It can be any character string. The character string must be enclosed in single quotation marks (') or double quotation marks (").

Form	Example
<String data>	'ABC' "IEEE488.2-1987"

- If a character string contains a double quotation mark ("), the double quotation mark is expressed as two consecutive quotation marks (""). This rule also applies to single quotation marks.
- A response message is always enclosed in double quotation marks (").
- <String data> is any character string. Therefore, the instrument assumes that the remaining program message units are part of the character string if no single (') or double quotation mark (") is encountered. As a result, no error is detected if a quotation mark is omitted.
- For information about the number of characters in a file name, see the Features Guide.

<Block Data>

<Block data> is any 8-bit data. It is only used in response messages on the instrument. The syntax is as follows:

Form	Example
#N<N-digit decimal number><data byte sequence>	#800000010ABCDEFGHIJ

- #N
  Indicates that the data is <block data>. “N” indicates the number of succeeding data bytes (digits) in ASCII code.
- <N-digit decimal number>
  Indicates the number of bytes of data (example: 00000010 = 10 bytes).
- <Data byte sequence>
  Expresses the actual data (example: ABCDEFGHIJ).
- Data is comprised of 8-bit values (0 to 255). This means that the ASCII code “0AH,” which stands for “NL,” can also be included in the data. Hence, care must be taken when programming the controller.

4.5 Synchronization with the Controller

Overlap Commands and Sequential Commands

There are two types of commands: overlap and sequential. The execution of one overlap command can start before the execution of the previous overlap command is completed.

If you specify V/div and send the next program message to query the result, the instrument always returns the most recent setting (5 V in this case).

```
:CHANnel1:VDIV 5V;VDIV?<PMT>
```

This is because the next command is forced to wait until the processing of `CHANnel1:VDIV` is completed. This type of command is called a sequential command.

Let us assume you send the next program message when you want to load a file and query the V/div value of the result.

```
:FILE:LOAD:SETup:EXECute "CASE1";:
CHANnel1:VDIV?
```

In this case, `CHANnel1:VDIV?` is executed before the loading of the file is completed, and the V/div value that is returned is the value before the file is loaded.

Overlapping refers to the act of executing the next command before the processing of the current command is completed, such as in the command `FILE:LOAD:SETup:EXECute "CASE1"`. A command that operates in this way is called an overlap command.

You can prevent overlapping by using the following methods.

Synchronizing to Overlap Commands

- **Using a *WAI Command**

  A `*WAI` command holds the subsequent commands until the overlap command is completed.

  Example
  ```
  :COMMunicate:OPSE #H0040;:
  FILE:LOAD:SETup:
  EXECute "CASE1";*WAI;:
  CHANnel1:VDIV?<PMT>
  ```

  The `COMMunicate:OPSE` command is used to select which command to apply `*WAI` to. Here, it is applied to the media access command.

  `*WAI` is executed before `CHANnel1:VDIV?`, so `CHANnel1:VDIV?` is not executed until the file loading is completed.

- **Using the COMMunicate:OVERlap command**

  The `COMMunicate:OVERlap` command enables (or disables) overlapping.

  Example
  ```
  :COMMunicate:OVERlap #HFFBF;:
  FILE:LOAD:SETup:
  EXECute "CASE1";:CHANnel1
  :VDIV?<PMT>
  ```

  `COMMunicate:OVERlap #HFFBF` enables overlapping for commands other than media access. Because overlapping of file loading is disabled, `FILE:LOAD:SETup:EXECute "CASE1"` operates in the same way as a sequential command.

  Thus, `CHANnel1:VDIV?` is not executed until file loading is completed.

Using the *OPC Command

The `*OPC` command sets the OPC bit, which is bit 0 in the standard event register (see page 6-4 for details), to 1 when the overlapping is completed.

Example
```
:COMMunicate:OPSE #H0040;
*ESE 1;*ESR?;*SRE 32;:FILE:
LOAD:SETup:
EXECute "CASE1";*OPC<PMT>
```
(Read the response to `*ESR?`)
(Wait for a service request)
```
:CHANnel1:VDIV?<PMT>
```

The `COMMunicate:OPSE` command is used to select which command to apply `*OPC` to. Here, it is applied to the media access command.

`*ESE 1` and `*SRE 32` indicate that a service request is only generated when the OPC bit is 1.

`*ESR?` clears the standard event register.

In the example above, `CHANnel1:VDIV?` is not executed until a service request is generated.

- **Using the *OPC? Query**
  The *OPC? query generates a response when an overlapping operation is completed.

  Example
  ```
  :COMMunicate:OPSE #H0040;:
  FILE:LOAD:SETup:
  EXECute "CASE1";*OPC?<PMT>
  ```
  (Read the response to *OPC?)
  ```
  :CHANnel1:VDIV?<PMT>
  ```
  The COMMunicate:OPSE command is used to select which command to apply *OPC? to. Here, it is applied to the media access command.
  Because *OPC? does not generate a response until the overlapping operation is completed, the file loading will have been completed by the time the response to *OPC? is read.

Note

Most commands are sequential commands. Overlap commands are indicated as such in chapter 5. All other commands are sequential commands.

Achieving Synchronization without Using Overlap Commands

Even with sequential commands, synchronization with non-communication events such as triggers is sometimes required to correctly query the measured data.
For example, if the following program message is transmitted to query waveform data acquired with the trigger mode set to single, the WAVeform:SEND? command may be executed regardless of whether or not the acquisition has been completed and may result in a command execution error.

```
TRIGger:MODE NORMal;:STARt;:WAVeform:
SEND?<PMT>
```

If this happens, you must use the following method to synchronize to the end of waveform acquisition.

- **Using the STATus:CONDition? query**
  STATus:CONDition? is used to query the contents of the condition register (see page 6-5 for details). You can determine whether or not waveform acquisition is in progress by reading bit 0 in the condition register. If the bit is 1, waveform acquisition is in progress. If the bit is 0, waveform acquisition is not in progress.

  Example
  ```
  TRIGger:MODE NORMal;:
  STARt<PMT>
  :STATus:CONDition?<PMT>
  ```
  (Read the response. If bit 0 is 1, return to the previous command.)
  ```
  :WAVeform:SEND?<PMT>
  ```
  WAVeform:SEND? is not executed until bit 0 in the condition register becomes 0.

- **Using the Extended Event Register**
  The changes in the condition register can be reflected in the extended event register (see page 6-5 for details).

  Example
  ```
  :STATus:FILTer1 FALL;:STATus:
  EESE 1;EESR?;*SRE 8;:TRIGger:
  MODE NORMal;:STARt<PMT>
  ```
  (Read the response to STATus:EESR?)
  (Wait for a service request)
  ```
  :WAVeform:SEND?<PMT>
  ```
  The STATus:FILTer1 FALL command sets the transition filter so that bit 0 in the extended event (FILTer1) is set to 1 when bit 0 in the condition register changes from 1 to 0.

  The STATus:EESE 1 command is used to only change the status byte based on bit 0 in the extended event register.
  The STATus:EESR? command is used to clear the extended event register.
  The *SRE 8 command is used to generate service requests based only on the changes in the extended event register bits.
  The WAVeform:SEND? command is not executed until a service request is generated.

- **Using the COMMunicate:WAIT command**
  The COMMunicate:WAIT command is used to wait for a specific event to occur.

  Example
  ```
  :STATus:FILTer1 FALL;:STATus:
  EESR?;:TRIGger:MODE NORMal
  <PMT>
  ```
  (Read the response to STATus:EESR?)
  ```
  :COMMunicate:WAIT 1;:
  WAVeform:SEND?<PMT>
  ```
  For a description of STATus:FILTer1 FALL and STATus:EESR?, see the previous section about the extended event register.
  The COMMunicate:WAIT 1 command specifies that the program will wait for bit 0 in the extended event register to be set to 1.
  WAVeform:SEND? is not executed until bit 0 in the extended event register becomes 1.

5.1 List of Commands

Command	Function	Page
ACQuire Group		
:ACQuire?	Queries all waveform acquisition settings.	5-57
:ACQuire		
:AVERage?	Queries all settings related to averaging and waveform acquisition counts.	5-57
:AVERage		
:COUNt	Sets or queries the attenuation constant of exponential averaging or the average count of simple averaging when in averaging mode.	5-57
:COUNt	Sets or queries the number of waveform acquisitions to perform in normal, envelope, and averaging modes.	5-57
:MODE	Sets or queries the waveform acquisition mode.	5-57
:RESolution	Sets or queries the high resolution mode on/off state.	5-57
:RLENgth	Sets or queries the record length.	5-57
:SAMPling	Sets or queries the sampling mode.	5-57
ANALysis Group		
:ANALysis?	Queries all analysis feature settings.	5-58
:ANALysis		
:AHIStogram<x>?	Queries all settings related to the histogram of the analysis feature.	5-58
:AHIStogram<x>		
:DISPlay	Sets or queries the waveform histogram display on/off state.	5-58
:HORizontal	Sets or queries the horizontal range of the waveform histogram.	5-58
:MEASure?	Queries all settings related to the waveform histogram measurement (including the mode on/off state).	5-58
:MEASure		
:MODE	Sets or queries the waveform histogram measurement mode.	5-58
:PARameter?	Queries all settings related to the waveform histogram measurement when the mode is set to Param.	5-58
:PARameter		
:ALL	Collectively turns on or off all the measurement items of the waveform histogram.	5-58
:<Parameter>?	Queries all settings related to the specified measurement item of the waveform histogram.	5-58
:<Parameter>		
:STATe	Sets or queries the on/off state of the specified measurement item of the waveform histogram.	5-59
:VALue?	Queries the automatically measured value of the specified measurement item of the waveform histogram.	5-59
:HPOSition<y>	Sets or queries the horizontal position of Cursor1 or Cursor2 of the waveform histogram.	5-59
:VPOSition<y>	Sets or queries the vertical position of Cursor1 or Cursor2 of the waveform histogram.	5-59
:MODE	Sets or queries the target axis of the waveform histogram.	5-59
:RANGe	Sets or queries the measurement source window of the waveform histogram.	5-59
:TRACe	Sets or queries the source waveform of the waveform histogram.	5-59
:VERTical	Sets or queries the vertical range of the waveform histogram.	5-60
:PANalyze?	Queries all power supply analysis settings.	5-60
:PANalyze		
:I2T?	Queries all Joule integral settings.	5-60
:I2T		
:MATH	Sets or queries the Joule integral waveform display on/off state.	5-60
:MEASure?	Queries all settings related to the automated measurement of the Joule integral.	5-60
:MEASure		
:I2T?	Queries Joule integral settings.	5-60
:I2T		
:COUNt?	Queries the continuous statistical processing count of the Joule integral.	5-60

Command	Function	Page
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries the Joule integral statistics.	5-60
:STATe	Sets or queries whether the Joule integral is to be measured.	5-60
:VALue?	Queries automatically measured value of the Joule integral.	5-61
:RANGe	Sets or queries the measurement source window.	5-61
:SCALe?	Queries all scaling settings.	5-61
:SCALe		
:CENTer	Sets or queries the center value for manual scaling.	5-61
:MODE	Sets or queries the scale mode.	5-61
:SENSitivity	Sets or queries the sensitivity for manual scaling.	5-61
:TRANge	Sets or queries the measurement time period.	5-61
:HARMonics?	Queries all harmonic analysis settings.	5-62
:HARMonics		
:CCLass?	Queries all settings related to harmonic analysis class C.	5-62
:CCLass		
:GETLambda	Sets the current power factor of harmonic analysis class C.	5-62
:LAMBda	Sets or queries the power factor of harmonic analysis class C.	5-62
:MAXCurrent	Sets or queries the fundamental current value of harmonic analysis class C.	5-62
:OPOWer	Sets or queries whether active power 25 W of harmonic analysis class C will be exceeded.	5-62
:CLASs	Sets or queries the applicable class of the DUT in harmonic analysis.	5-62
:DCLass?	Queries all settings related to harmonic analysis class D.	5-62
:DCLass		
:POWer	Sets or queries the power value of harmonic analysis class D.	5-62
:DMODe	Sets or queries the display mode of harmonic analysis.	5-63
:DETail?	Queries all settings related to the harmonic analysis result list.	5-63
:DETail		
:DISPlay	Sets or queries the display mode of the harmonic analysis result list.	5-63
:LIST		
:ITEM?	Queries all items that will be displayed in the harmonic analysis result list.	5-63
:VALue?	Queries all the data for the specified analysis number in the harmonic analysis result list.	5-63
:GROuping	Sets or queries the harmonic analysis grouping.	5-63
:RMS	Queries the rms value harmonic analysis.	5-63
:SPOint	Sets or queries the computation start point of harmonic analysis.	5-63
:SVOLtage	Sets or queries the supply voltage of harmonic analysis.	5-64
:THD	Queries the harmonic distortion factor of harmonic computation.	5-64
:SETup?	Queries all power supply analysis input settings.	5-64
:SETup		
:ADESkew	Executes auto deskewing for power supply analysis.	5-64
:I?	Queries all current input channel settings for power supply analysis.	5-64
:I		
:DESKew	Sets or queries auto deskewing of the current input channel for power supply analysis.	5-64
:INPut	Sets or queries the current input channels for power supply analysis.	5-64
:PROBe	Queries all current-to-voltage conversion ratio settings of the probe connected to the current input channel for power supply analysis.	5-64
:U?	Queries all voltage input channel settings for power supply analysis.	5-64
:U		
:DESKew	Sets or queries auto deskewing of the voltage input channel for power supply analysis.	5-64
:INPut	Sets or queries the voltage input channels for power supply analysis.	5-65
:PROBe	Sets or queries the probe attenuation of the voltage input channels for power supply analysis.	5-65
:RTRace	Sets or queries the deskewing source trace for power supply analysis.	5-65
:SOA?	Queries all settings related to the XY display for safe operating area analysis.	5-65
:SOA		
:CURSor?	Queries all settings related to the cursor measurement on the XY display for safe operating area analysis.	5-65
:CURSor		
:X<x>?	Queries all settings related to the horizontal cursors on the XY display for safe operating area analysis.	5-65

Command	Function	Page
:X<x>		
:POSition	Sets or queries the horizontal cursor positions on the XY display for safe operating area analysis.	5-65
:VALue?	Queries the horizontal cursor voltages on the XY display for safe operating area analysis.	5-65
:Y<x>?	Queries all settings related to the vertical cursors on the XY display for safe operating area analysis.	5-65
:Y<x>		
:POSition	Sets or queries the vertical cursor positions on the XY display for safe operating area analysis.	5-65
:VALue?	Queries the vertical cursor voltages on the XY display for safe operating area analysis.	5-66
:MODE	Sets or queries the automated measurement mode on the XY display for safe operating area analysis.	5-66
:TRANge	Sets or queries the VT waveform range to show on the XY display for safe operating area analysis.	5-66
:VTDisplay	Sets or queries the on/off state of the VT waveform display on the XY display for safe operating area analysis.	5-66
:SWLoss?	Queries all switching loss settings.	5-66
:SWLoss		
:CCALc	Sets or queries the total loss equation.	5-66
:DPROximal?	Queries all distal, mesial, and proximal settings.	5-66
:DPROximal		
:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-67
:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-67
:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-67
:ILEVel	Sets or queries the current level for determining the loss zero period of total loss.	5-67
:INDicator	Sets or queries the total loss measurement location indicator.	5-67
:MATH	Sets or queries the power waveform display on/off state.	5-67
:MEASure?	Queries all settings related to the automated measurement of power supply parameters.	5-67
:MEASure		
:<Parameter>?	Queries the setting of a power supply analysis parameter.	5-67
:<Parameter>		
:COUNt?	Queries the continuous statistical processing count of the power supply analysis parameter.	5-68
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a power supply analysis parameter.	5-68
:STATe	Sets or queries the on/off state of the power supply analysis parameter.	5-68
:VALue?	Queries the automatically measured value of a power supply parameter.	5-68
:CONTinuous?	Queries all settings related to the normal statistical processing of automatically measured total loss.	5-68
:CONTinuous		
:RESTart	Restarts the normal statistical processing of automatically measured total loss.	5-68
:TLCHange	Sets or queries whether the normal statistical processing of automatically measured total loss is to be restarted.	5-69
:CYCLe		
:ABORt	Aborts the execution of the cyclic statistical processing of automatically measured total loss.	5-69
:EXECute	Executes the cyclic statistical processing of automatically measured total loss.	5-69
:HISTory		
:ABORt	Aborts the execution of the statistical processing of the history waveform of automatically measured total loss.	5-69
:EXECute	Executes the statistical processing of the history waveform of automatically measured total loss.	5-69
:MODE	Sets or queries the statistical processing type of automatically measured total loss.	5-69
:ZLINkage	Sets or queries the linkage (on/off) between the result numbers and zoom positions of the results of automatically measured total loss.	5-69
:METhod	Sets or queries the calculation method for high and low points.	5-69
:RANGe	Sets or queries the measurement source window.	5-69

5.1 List of Commands

Command	Function	Page
:RDS	Sets or queries the total loss on-resistance.	5-70
:SCALe?	Queries all scaling settings.	5-70
:SCALe		
:CENTer	Sets or queries the center value for manual scaling.	5-70
:MODE	Sets or queries the scale mode.	5-70
:SENSitivity	Sets or queries the sensitivity for manual scaling.	5-70
:TRANge	Sets or queries the measurement time period.	5-70
:ULEVel	Sets or queries the voltage level for determining the loss calculation period of total loss.	5-70
:UNIT	Sets or queries the watthour unit.	5-70
:VCE	Sets or queries the collector-emitter saturation voltage of total loss.	5-71
:TYPE	Sets or queries the power supply analysis type.	5-71
:PMEAsure<x>?	Queries all power measurement settings.	5-71
:PMEAsure<x>		
:CYCMode	Sets or queries power measurement cycle mode.	5-71
:CYCTrace	Sets or queries the source waveform of power measurement cycle mode.	5-71
:IDPRoximal?	Queries all distal, mesial, and proximal settings.	5-71
:IDPRoximal		
:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-71
:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-71
:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-71
:IMEThod	Sets or queries the high and low points.	5-72
:INDicator	Sets or queries the measurement location indicator.	5-72
:MEASure?	Queries all settings related to the automated measurement of power measurement parameters.	5-72
:MEASure		
:<Parameter>?	Queries the power measurement parameter settings.	5-72
:<Parameter>		
:COUNt?	Queries the normal statistical processing count of the power measurement parameter.	5-72
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries the statistical value of a power measurement parameter.	5-72
:STATe	Sets or queries the on/off state of a power measurement parameter.	5-72
:VALue?	Queries the automatically measured power measurement parameter value.	5-73
:ALL	Collectively turns on or off the power measurement parameters.	5-73
:CONTinuous?	Queries all settings related to the normal statistical processing of automated power measurement.	5-73
:CONTinuous		
RESTart	Restarts the normal statistical processing of automated power measurement.	5-73
:TLCHange	Sets or queries whether the normal statistical processing of automated power measurement is to be restarted.	5-73
:CYCLe?	Queries all settings related to the cyclic statistical processing of automated power measurement.	5-73
:CYCLe		
:ABORt	Aborts the execution of the cyclic statistical processing of automated power measurement.	5-73
:EXECute	Executes the cyclic statistical processing of automated power measurement.	5-73
:TRACe	Sets or queries the source waveform of the cyclic statistical processing of automated power measurement.	5-73
:HISTory		
:ABORt	Aborts the statistical processing of the history waveform of automated power measurement.	5-73
:HISTory		
:EXECute	Executes the statistical processing of the history waveform of automated power measurement.	5-74
:MODE	Sets or queries the statistical processing type of automated power measurement.	5-74
:ZLINkage	Sets or queries the linkage (on/off) between the result numbers and zoom positions of the results of automated power measurement.	5-74
:MODE	Sets or queries the on/off state of power measurement.	5-74
:RANGe	Sets or queries the measurement source window.	5-74
:SETup?	Queries all power measurement input settings.	5-74
:SETup		

Command	Function	Page
:ADESkew	Executes auto deskewing for power measurement.	5-74
:I?	Queries all current input channel settings for power measurement.	5-74
:I		
:DESKew	Sets or queries auto deskewing of the current input channel for power measurement.	5-74
:PROBe	Queries all current-to-voltage conversion ratio settings of the probe connected to the current input channel for power measurement.	5-75
:RTRace	Sets or queries the deskewing source trace for power measurement.	5-75
:U?	Queries all voltage input channel settings for power measurement.	5-75
:U		
:DESKew	Sets or queries auto deskewing of the voltage input channel for power measurement.	5-75
:PROBe	Queries all voltage-to-voltage conversion ratio settings of the probe connected to the voltage input channel for power measurement.	5-75
:TRANge	Sets or queries the measurement time period.	5-75
:UDPRoximal?	Queries all distal, mesial, and proximal settings.	5-75
:UDPRoximal		
:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-75
:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-76
:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-76
:UMEThod	Sets or queries the high and low points.	5-76
:UNIT	Sets or queries the watthour unit.	5-76
:USER<y>?	Queries all automatic measurement settings for a Calc item.	5-76
:USER<y>		
:COUNt?	Queries the statistical processing count of the automatically measured value of a Calc item.	5-76
:DEFine	Sets or queries the expression for the automatically measured value of a Calc item.	5-76
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value that is calculated on the automatically measured value of a Calc item.	5-76
:NAME	Sets or queries the name of a Calc item.	5-77
:STATe	Sets or queries the on/off status of automated measurement of a Calc item.	5-77
:UNIT	Sets or queries the unit of a Calc item.	5-77
:VALue?	Queries the automatically measured value of a Calc item.	5-77
:WAIT?	Waits for the completion of automated measurement with a timeout.	5-77

ASETup Group

:ASETup		
:EXECute	Executes auto setup.	5-78
:UNDO	Undoes auto setup.	5-78

CALibrate Group

:CALibrate?	Queries all calibration settings.	5-79
:CALibrate		
[:EXECute]	Executes calibration.	5-79
:MODE	Sets or queries the auto calibration on/off state.	5-79

CHANnel Group

:CHANnel<x>?	Queries all vertical axis settings of a channel.	5-80
:CHANnel<x>		
:ASCale		
[:EXECute]	Executes auto scaling of a channel.	5-80
:BWIDth	Sets or queries the input filter of the specified channel.	5-80
:COUPling	Sets or queries the input coupling of the specified channel.	5-80
:DESKew	Sets or queries the deskewing of the specified channel.	5-80
:DISPlay	Sets or queries the display on/off state of the specified channel.	5-80
:INVert	Sets or queries the on/off state of the invert mode (waveform inversion display) .	5-80
:LABel?	Queries all settings related to the waveform label of the specified channel.	5-80
:LABel		

Command	Function	Page
:DEFine	Sets or queries the waveform label of a channel.	5-80
:DISPlay	Sets or queries the display on/off state of the specified channel's waveform label.	5-81
:LSCale?	Queries all the linear scaling settings of the specified channel.	5-81
:LSCale		
:AVALue	Sets or queries the scaling coefficient A.	5-81
:BVALue	Sets or queries offset B.	5-81
:MODE	Sets or queries the on/off state of linear scaling.	5-81
:UNIT	Sets or queries a unit that is attached to linear scaling results.	5-81
:OFFSet	Sets or queries the offset voltage of the specified channel.	5-81
:POSition	Sets or queries the vertical position of the specified channel.	5-81
:PROBe?	Queries all settings related to the probe attenuation of the specified channel.	5-81
:PROBe		
:DZCalibrate	Executes demagnetization and zero calibration of the specified channel.	5-81
[:MODE]	Sets or queries the probe attenuation of the specified channel.	5-82
:PZCalibrate	Executes zero calibration of the specified channel's current probe.	5-82
:VARiable	Sets or queries the specified channel's voltage scale in 0.01 V/div steps.	5-82
:VDIV	Sets or queries the specified channel's voltage scale (V/div).	5-82

CLEar Group

:CLEar	Clears traces.	5-83

COMMunicate Group

:COMMunicate?	Queries all communication settings.	5-84
:COMMunicate		
:HEADer	Sets or queries whether a header is added to the response to a query (example with header: "CHANNEL1:PROBE:MODE 10"; example without header: "10").	5-84
:LOCKout	Sets/clears local lockout.	5-84
:OPSE	Sets or queries the overlap command that is used by the *OPC, *OPC?, and *WAI commands.	5-84
:OPSR?	Queries the operation pending status register.	5-84
:OVERlap	Sets or queries the commands that operate as overlap commands.	5-84
:REMote	Sets the instrument to remote or local mode. On is remote mode.	5-84
:VERBose	Sets or queries whether the response to a query is returned fully spelled out (e.g., CHANNEL1:PROBE:MODE 10) or in its abbreviated form (e.g., CHAN:PROB 10).	5-85
:WAIT	Waits for a specified extended event to occur.	5-85
:WAIT?	Creates the response that is returned when a specified extended event occurs.	5-85

CURSor Group

:CURSor?	Queries all cursor measurement settings.	5-86
:CURSor		
[:TY]?	Queries all cursor settings.	5-86
[:TY]		
:DEGRee?	Queries all angle cursor settings.	5-86
:DEGRee		
:ALL	Collectively turns on or off the measured values of the angle cursor on the VT display.	5-86
:D<x>?	Queries all angle cursor's angle settings.	5-86
:D<x>		
:STATe	Sets or queries the on/off state of the angle cursor's angle.	5-86
:VALue?	Queries the the angle cursor's angle.	5-86
:DD?	Queries all settings related to the angle difference between the angle cursors.	5-86
:DD		
:STATe	Sets or queries the on/off state of the angle difference ΔD value between the angle cursors.	5-86
:VALue?	Queries the angle difference ΔD value between the angle cursors.	5-86
:DV?	Queries all settings related to the ΔV value between the angle cursors.	5-86
:DV		

Command	Function	Page
:STATe	Sets or queries the on/off state of the ΔV value between the angle cursors.	5-86
:VALue?	Queries the ΔV value between the angle cursors.	5-86
:JUMP	Moves the angle cursor on the VT display on to the specified zoom waveform.	5-87
:POSition<x>	Sets or queries the angle cursor position.	5-87
:REFerence<x>	Sets or queries the start point (Reference1) or end point (Reference2) of the reference angle.	5-87
:RVALue	Sets or queries the reference angle.	5-87
:TRACe	Sets or queries the source waveform of the angle cursor.	5-87
:UNIT	Sets or queries the angle cursor unit.	5-87
:V<x>?	Queries all angle cursor's voltage settings.	5-87
:V<x>		
:STATe	Sets or queries the on/off state of the angle cursor's voltage.	5-87
:VALue?	Queries the angle cursor's voltage.	5-87
:HORizontal?	Queries all ΔV cursor settings.	5-87
:HORizontal		
:ALL	Collectively turns on or off the measured values of the ΔV cursor on the VT display.	5-88
:DV?	Queries all settings related to the vertical value between the ΔV cursors.	5-88
:DV		
:STATe	Sets or queries the on/off state of the vertical value between the ΔV cursors.	5-88
:VALue?	Queries the vertical value between the ΔV cursors.	5-88
:POSition<x>	Sets or queries the ΔV cursor position.	5-88
:TRACe	Sets or queries the source waveform of the ΔV cursor.	5-88
:V<x>?	Queries all settings related to the ΔV cursor's vertical axis.	5-88
:V<x>		
:STATe	Sets or queries the on/off state of the ΔV cursor's vertical value.	5-88
:VALue?	Queries the ΔV cursor's vertical value.	5-88
:MARKer?	Queries all marker cursor settings.	5-88
:MARKer		
:FORM	Sets or queries the display format of the marker cursor.	5-88
:M<x>?	Queries all settings related to the specified marker cursor.	5-89
:M<x>		
:ALL	Collectively turns on or off the measured values of the marker cursor on the VT display.	5-89
:DT<y>?	Queries all settings related to the time value between the marker cursors.	5-89
:DT<y>		
:STATe	Sets or queries the on/off state of the time value between the marker cursors.	5-89
:VALue?	Queries the time value between the marker cursors.	5-89
:DV<y>?	Queries all settings related to the vertical value between the marker cursors.	5-89
:DV<y>		
:STATe	Sets or queries the on/off state of the vertical value between the marker cursors.	5-89
:VALue?	Queries the vertical value between the marker cursors.	5-89
:JUMP	Moves the marker cursor on the VT display on to the specified zoom waveform.	5-89
:POSition	Sets or queries the marker cursor's time value.	5-89
:T?	Queries all settings related to the marker cursor's time value.	5-90
:T		
:STATe	Sets or queries the on/off state of the marker cursor's time value.	5-90
:VALue?	Queries the marker cursor's time value.	5-90
:TRACe	Sets or queries the source waveform of the marker cursor.	5-90
:V?	Queries all settings related to the marker cursor's vertical value.	5-90
:V		
:STATe	Sets or queries the on/off state of the marker cursor's vertical value.	5-90
:VALue?	Queries the marker cursor's vertical value.	5-90
:TYPE	Sets or queries the cursor type.	5-90
:VERTical?	Queries all ΔT cursor settings.	5-90
:VERTical		
:ALL	Collectively turns on or off the measured values of the ΔT cursor on the VT display.	5-90
:DT?	Queries all settings related to the time value between the ΔT cursors.	5-90
:DT		

Command	Function	Page
:STATe	Sets or queries the on/off state of the time value between the ΔT cursors.	5-90
:VALue?	Queries the time value between the ΔT cursors.	5-90
:DV?	Queries all settings related to the vertical value between the ΔT cursors.	5-91
:DV		
:STATe	Sets or queries the on/off state of the vertical value between the ΔT cursors.	5-91
:VALue?	Queries the vertical value between the ΔT cursors.	5-91
:JUMP	Moves the vertical cursor on the VT display on to the specified zoom waveform.	5-91
:PERDt?	Queries all settings related to the 1/ΔT value between the ΔT cursors.	5-91
:PERDt		
:STATe	Sets or queries the on/off state of the 1/ΔT value between the ΔT cursors.	5-91
:VALue?	Queries the 1/ΔT value between the ΔT cursors.	5-91
:POSition<x>	Sets or queries the ΔT cursor position.	5-91
:T<x>?	Queries all settings related to the ΔT cursor's time value.	5-91
:T<x>		
:STATe	Sets or queries the on/off state of the ΔT cursor's time value.	5-91
:VALue?	Queries the ΔT cursor's time value.	5-91
:TRACe	Sets or queries the source waveform of the ΔT cursor.	5-91
:V<x>?	Queries all settings related to the ΔT cursor's vertical value.	5-92
:V<x>		
:STATe	Sets or queries the on/off state of the ΔT cursor's vertical value.	5-92
:VALue?	Queries the ΔT cursor's vertical value.	5-92

DISPlay Group

:DISPlay?	Queries all display settings.	5-93
:DISPlay		
:ACCumulate?	Queries all accumulated waveform display settings.	5-93
:ACCumulate		
:MODE	Sets or queries the accumulation mode.	5-93
:PERSistence	Sets or queries the accumulation time.	5-93
:COLor?	Queries all waveform color settings.	5-93
:COLor		
:{CHANnel<x>|MATH<x>}	Sets or queries a waveform color.	5-93
:LSTate	Sets or queries the logic waveform state display color.	5-93
:SERialbus<x>?	Queries the color settings of all trends of the specified serial bus.	5-93
:SERialbus<x>		
:TRENd<y>	Sets or queries the serial bus trend color.	5-93
:FORMat	Sets or queries the display format.	5-93
:FGRid	Sets or queries whether the fine grid is displayed.	5-94
:GRATicule	Sets or queries the graticule (grid).	5-94
:INTENsity?	Queries all settings related to the intensity of display items.	5-94
:INTENsity		
[:WAVeform]	Sets or queries the waveform intensity.	5-94
:{CURSor|GRID|MARKer|ZBOX}	Sets or queries the intensity of a display item.	5-94
:INTERpolate	Sets or queries the interpolation method.	5-94
:MAPPing	Sets or queries the split screen waveform mapping mode.	5-94
:SMAPping?	Queries all waveform mappings to the split screen.	5-94
:SMAPping		
:{CHANnel<x>|MATH<x>}	Sets or queries the specified waveform mapping to the split screen.	5-94
:SVALue	Sets or queries the on/off status of the scale value display.	5-94

FFT Group

:FFT<x>?	Queries all FFT analysis settings.	5-95
:FFT<x>		
:AVERage?	Queries all settings related to the averaging of FFT analysis.	5-95
:AVERage		
:EWEight	Sets or queries the attenuation constant of exponential averaging of FFT analysis.	5-95
:DATA?	Queries all the information related to the FFT waveform data that will be sent.	5-95
:DATA		

Command	Function	Page
:BYTeorder	Sets or queries the transmission byte order when the FFT waveform data format is binary.	5-95
:END	Sets or queries the end point of the FFT waveform data that will be sent.	5-95
:FORMat	Sets or queries the format of the FFT waveform data that will be sent.	5-95
:LENGth?	Queries the total number of data points of the FFT waveform that will be sent.	5-95
:SEND?	Queries the FFT waveform data.	5-95
:STARt	Sets or queries the start point of the FFT waveform data that will be sent.	5-95
:DISPlay	Sets or queries whether FFT analysis is to be performed.	5-96
:HORizontal?	Queries all horizontal axis settings of FFT analysis.	5-96
:HORizontal		
:CSPan?	Queries all settings related to the center point and span of the horizontal axis in FFT analysis.	5-96
:CSPan		
:CENTer	Sets or queries the center value of the horizontal axis in FFT analysis.	5-96
:SPAN	Sets or queries the span of the horizontal axis in FFT analysis.	5-96
:LRIGht?	Queries all settings related to the left and right edges of the horizontal axis in FFT analysis.	5-96
:LRIGht		
:RANGe	Sets or queries the range of the left and right edges of the horizontal axis in FFT analysis.	5-96
:MODE	Sets or queries the horizontal axis mode of FFT analysis.	5-96
:LENGth	Sets or queries the number of FFT points of FFT analysis.	5-96
:MEASure?	Queries all automated measurement settings of FFT analysis.	5-96
:MEASure		
:MARKer?	Queries all settings related to the marker cursor measurement of FFT analysis.	5-96
:MARKer		
[:BASic]?	Queries all settings related to the basic marker cursor items of FFT analysis.	5-96
[:BASic]		
:ALL	Collectively turns on or off all the basic marker cursor items of FFT analysis.	5-97
:DFRequency?	Queries all settings related to the frequency between marker cursors of FFT analysis.	5-97
:DFRequency		
:STATe	Sets or queries the on/off state of the frequency between marker cursors of FFT analysis.	5-97
:VALue?	Queries the frequency between marker cursors of FFT analysis.	5-97
:DV?	Queries all settings related to the level between marker cursors of FFT analysis.	5-97
:DV		
:STATe	Sets or queries the on/off state of the level between marker cursors of FFT analysis.	5-97
:VALue?	Queries the level between marker cursors of FFT analysis.	5-97
:FREQuency<y>?	Queries all settings related to the specified marker cursor's frequency of FFT analysis.	5-97
:FREQuency<y>		
:STATe	Sets or queries the on/off state of the specified marker cursor's frequency of FFT analysis.	5-97
:VALue?	Queries the specified marker cursor's frequency of FFT analysis.	5-98
:POSition<y>	Sets or queries the specified marker cursor's position of FFT analysis.	5-98
:V<y>?	Queries all settings related to the specified marker cursor's level of FFT analysis.	5-98
:V<y>		
:STATe	Sets or queries the on/off state of the specified marker cursor's level of FFT analysis.	5-98
:VALue?	Queries the specified marker cursor's level of FFT analysis.	5-98
:MODE	Sets or queries the automated measurement mode of FFT analysis.	5-98
:PEAK?	Queries all settings related to the peak value measurement of FFT analysis.	5-98
:PEAK		
:DETail?	Queries all settings related to the FFT analysis result list.	5-98
:DETail		
:LIST		

Command	Function	Page
:ITEM?	Queries all items that will be displayed in the FFT analysis result list.	5-98
:MAXPeak		
:EXECute	Moves to the maximum peak value in the FFT analysis results.	5-98
:NUMBer	Sets or queries the analysis number in the FFT analysis result list.	5-99
:VALue?	Queries all the data for the specified analysis number in the FFT analysis result list.	5-99
:EXCursion	Sets or queries the difference between peak and valley of the peak value of FFT analysis.	5-99
:FVALue?	Queries the peak frequency of FFT analysis.	5-99
:THReshold	Sets or queries the threshold level of the peak value of FFT analysis.	5-99
:VVALue?	Queries the peak value of FFT analysis.	5-99
:MODE	Sets or queries the waveform display mode of FFT analysis.	5-99
:RANGe	Sets or queries the measurement source window of FFT analysis.	5-99
:RPOSition	Sets or queries the center zoom point on the vertical axis in FFT analysis.	5-99
:TRACe	Sets or queries the source waveform of FFT analysis.	5-100
:TYPE	Sets or queries the spectrum of FFT analysis.	5-100
:UNIT?	Queries all FFT computation unit settings.	5-100
:UNIT		
[:DEFine]	Sets or queries the FFT computation unit.	5-100
:MODE	Sets or queries whether an FFT computation unit will be attached automatically or manually.	5-100
:VERTical?	Queries all vertical axis settings of FFT analysis.	5-100
:VERTical		
:LEVel	Sets or queries the vertical axis display position of FFT analysis.	5-100
:MODE	Sets or queries the vertical axis mode of FFT analysis.	5-100
:SENSitivity	Sets or queries the vertical axis sensitivity of FFT analysis.	5-101
:VTDisplay	Sets or queries the on/off state of the VT waveform display of FFT analysis.	5-101
:WINDow	Sets or queries the window function of FFT analysis.	5-101

FILE Group

:FILE?	Queries all data storage settings.	5-102
:FILE		
:COPY		
:ABORt	Aborts file copying.	5-102
:CDIRectory	Changes the file copy destination directory.	5-102
:DRIVe	Sets the storage device to copy files to.	5-102
[:EXECute]	Executes file copying. This is an overlap command.	5-102
:PATH?	Queries the file copy destination directory.	5-102
:DELete		
:{AHIStogram|ASCii|BINary|BMP|FFT|HLISt|JPEG|MEASure|PNG|SBUS|SETup|SNAP|ZPOLygon|ZWAVe}		
[:EXECute]	Deletes various types of data files. This is an overlap command.	5-102
[:DIRectory]		
:CDIRectory	Changes the current directory.	5-102
:DRIVe	Sets the current storage device.	5-102
:FREE?	Queries the free space on the current storage device that is being operated on in bytes.	5-102
:MDIRectory	Creates a directory in the current location. This is an overlap command.	5-102
:PATH?	Queries the current directory.	5-103
:LOAD		
:BINary		
:ABORt	Aborts the loading of waveform data.	5-103
[:EXECute]	Executes the loading of waveform data. This is an overlap command.	5-103
:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}		
:ABORt	Aborts the loading of various types of data.	5-103
[:EXECute]	Executes the loading of various types of data. This is an overlap command.	5-103
:MOVE		
:ABORt	Aborts file moving.	5-103
:CDIRectory	Changes the file move destination directory.	5-103

Command	Function	Page
:DRIVe	Sets the storage device to move files to.	5-103
[:EXECute]	Executes file moving. This is an overlap command.	5-103
:PATH?	Queries the file move destination directory.	5-103
:PROTect		
[:EXECute]	Turns on or off the file protection.	5-103
:REName		
[:EXECute]	Renames the file.	5-103
:SAVE?	Queries all file save settings.	5-103
:SAVE		
:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|SBUS|ZWAVe}?	Queries all the settings related to the saving of a specific type of file.	5-103
:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}		
:ABORt	Aborts the saving of a specific type of file.	5-104
[:EXECute]	Executes the saving of a specific type of file. This is an overlap command.	5-104
:{AHIStogram|FFT|HARMonics|SBUS|ZWAVe}		
:SELect	Sets or queries the area for saving a specific type of file.	5-104
:ANAMing	Sets or queries the on/off state of the auto generation of the name of the file to be saved.	5-104
:{ASCii|BINary}		
:COMPression	Sets or queries the save compression method for a specific type of file.	5-104
:HISTory	Sets or queries the save method for the specified history waveform.	5-104
:LENGth	Sets or queries the number of data points to save when compressing or sampling (decimating) various types of data.	5-104
:RANGe	Sets or queries the window for saving a specific type of data.	5-104
:TRACe	Sets or queries the waveform for saving a specific type of data.	5-105
:ASCii		
:TINFormation	Sets or queries whether waveform data is saved with time information (ON) or without it (OFF).	5-105
:COMMent	Sets or queries the comment for the data to be saved.	5-105
:FFT		
:FINFormation	Sets or queries whether FFT data is saved with frequency information (ON) or without it (OFF).	5-105
:NAME	Sets or queries the file name for the data to be saved.	5-105
:HISTory	Sets or queries the history for which serial bus data will be saved.	5-105

GONogo Group

:GONogo?	Queries all GO/NO-GO determination settings.	5-106
:GONogo		
:ABORt	Aborts GO/NO-GO determination.	5-106
:ACTion?	Queries all settings for the action that is performed when the judgment is NO-GO and the reference for the judgment.	5-106
:ACTion		
:BUZZer	Sets or queries whether a warning sound is to be generated when the judgment is NO-GO.	5-106
:HCOPy	Sets or queries whether a hard copy is output to the built-in printer (option) or an external printer when the judgment is NO-GO.	5-106
:MAIL?	Queries all settings related to the mail notification that is made when the judgment is NO-GO.	5-106
:MAIL		
:COUNt	Sets or queries the upper limit to the number of mail notifications made when the judgment is NO-GO.	5-106
:MODE	Sets or queries whether a mail notification is to be made when the judgment is NO-GO.	5-106
:SAVE	Sets or queries whether the event is to be saved to the storage device when the judgment is NO-GO.	5-106
:COUNt?	Queries the number of GO/NO-GO determinations that were performed.	5-106
:EXECute	Executes GO/NO-GO determination. This is an overlap command.	5-106
:LOGic	Sets or queries the GO/NO-GO determination type.	5-106
:NGCount?	Queries the GO/NO-GO determination NO-GO count.	5-106

Command	Function	Page
:NGStopcount	Sets or queries the number of NO-GO results.	5-107
:STOPcount	Sets or queries the number of determination results acquired.	5-107
:WAIT?	Waits for the completion of GO/NO-GO determination with a timeout.	5-107
[:ZPARameter]?	Queries all zone or parameter determination settings.	5-107
[:ZPARameter]		
:NUMBer<x>?	Queries all settings related to the zone or parameter determination conditions.	5-107
:NUMBer<x>		
:CAUSe?	Queries whether the specified waveform parameter for zone or parameter determination is the cause of a NO-GO judgment.	5-107
:CONDition	Sets or queries the specified waveform parameter's reference condition for zone or parameter determination.	5-107
:MODE	Sets or queries the mode of the specified condition.	5-107
:PARameter?	Queries all the parameter settings of the specified condition.	5-107
:PARameter		
:ITEM	Sets or queries the specified waveform parameter’s item for parameter determination.	5-108
:LIMit	Sets or queries the specified waveform parameter’s upper and lower limits for parameter determination.	5-108
:TRACe	Sets or queries the specified waveform parameter's source waveform for parameter determination.	5-108
:VALue?	Queries the measured value of the specified waveform parameter for parameter determination.	5-108
:POLYgon?	Queries all polygonal zone determination settings.	5-108
:POLYgon		
:HPOSition	Sets or queries the horizontal position used in polygonal zone determination.	5-109
:RANGe	Sets or queries the source window used in polygonal zone determination.	5-109
:TRACe	Sets or queries the source waveform used in polygonal zone determination.	5-109
:VPOSition	Sets or queries the vertical position used in polygonal zone determination.	5-109
:ZNUMber	Sets or queries the zone number used in polygonal zone determination.	5-109
:RECTangle?	Queries all waveform rectangular zone determination settings.	5-109
:RECTangle		
:HORizontal	Sets or queries the horizontal position of the rectangle used in rectangular zone determination.	5-110
:RANGe	Sets or queries the source window of the rectangle used in rectangular zone determination.	5-110
:TRACe	Sets or queries the source waveform of the rectangle used in rectangular zone determination.	5-110
:VERTical	Sets or queries the vertical position of the rectangle used in rectangular zone determination.	5-110
:WAVE?	Queries all waveform zone determination settings.	5-110
:WAVE		
:EDIT		
:EXIT	Exits from the waveform zone edit menu.	5-110
:NEW	Sets the base waveform in waveform zone editing.	5-110
:PART	Executes partial editing in waveform zone editing.	5-111
WHOLe	Sets the entire waveform zone in waveform zone editing.	5-111
:RANGe	Sets or queries the source window used in waveform zone determination.	5-111
:TRACe	Sets or queries the source waveform used in waveform zone determination.	5-111
:TRANge	Sets or queries the determination period used in waveform zone determination.	5-111
:ZNUMber	Sets or queries the zone number used in waveform zone determination.	5-111

HCOPy Group

:HCOPy?	Queries all screen capture data output settings.	5-112
:HCOPy		
:ABORt	Stops data output and paper feeding.	5-112
:DIRection	Sets or queries the data output destination.	5-112
:EXECute	Executes data output.	5-112
:EXTPrinter?	Queries all external printer output settings.	5-112
:EXTPrinter		
:MODE	Sets or queries the normal or hard copy setting for outputting to the external printer.	5-112

Command	Function	Page
:TONE	Sets or queries the external printer output color.	5-112
:MULTitarget?	Queries all multi target feature settings.	5-112
:MULTitarget		
:EXTPrinter	Sets or queries whether the multi target feature will send output to the external printer.	5-112
:PRINter	Sets or queries whether the multi target feature will send output to the built-in printer.	5-112
:NETPrinter	Sets or queries whether the multi target feature will send output to a network printer.	5-112
:FILE	Sets or queries whether the multi target feature will send output to a file.	5-113
:WAVeform	Sets or queries whether the multi target feature will send output to a waveform file.	5-113
:NETPrint?	Queries all network printer output settings.	5-113
:NETPrint		
:MODE	Sets or queries the normal or hard copy setting for outputting to the network printer.	5-113
:TONE	Sets or queries the network printer output color.	5-113
:TYPE	Sets or queries the network printer output command type.	5-113
:PRINter?	Queries all built-in printer output settings.	5-113
:MODE	Sets or queries the short or hard copy setting for outputting to the built-in printer.	5-113
:REPort	Sets or queries whether additional information is to be output to the built-in printer.	5-113

HISTory Group

:HISTory?	Queries all of the settings for the history feature.	5-114
:HISTory		
:AVERage	Sets or queries the history waveform highlight display mode.	5-114
:DISPlay	Sets or queries the history record start and end numbers that will be displayed.	5-114
:DMODe	Sets or queries the history waveform display mode.	5-114
:RECord	Sets or queries the history waveform source record.	5-114
:RECord?		
MINimum	Queries the smallest record number of the history waveform.	5-114
:REPLay?	Queries all history waveform replay settings.	5-114
:REPLay		
:JUMP	Moves to the specified record number in the history waveform.	5-114
:SPEed	Sets or queries the history waveform replay speed.	5-114
:STARt	Starts replaying the history waveform in the specified direction.	5-114
:STOP	Stops replaying the history waveform.	5-114
[:SEARch]?	Queries all history waveform search settings.	5-114
[:SEARch]		
:ABORt	Aborts the search.	5-115
:EXECute	Executes a search. This is an overlap command.	5-115
:LOGic	Sets or queries the history waveform search logic.	5-115
:NUMBer<x>?	Queries all search condition settings.	5-115
:NUMBer<x>		
:CONDition	Sets or queries the reference condition of the specified search condition.	5-115
:MODE	Sets or queries the mode of the specified search condition.	5-115
:PARameter?	Queries all parameter search settings.	5-115
:PARameter		
:ITEM	Sets or queries the specified waveform parameter's item for parameter searching.	5-115
:LIMit	Sets or queries the specified waveform parameter's upper and lower limits for parameter searching.	5-115
:TRACe	Sets or queries the specified waveform parameter's source waveform for parameter searching.	5-116
VALue?	Queries the measured value of the specified waveform parameter for parameter searching.	5-116
:NUMBer<x>		
:POLYgon?	Queries all polygonal zone search settings.	5-116
:POLYgon		
:HPOSition	Sets or queries the horizontal position used in polygonal zone searching.	5-116

Command	Function	Page
:RANGe	Sets or queries the source window used in polygonal zone searching.	5-116
:TRACe	Sets or queries the source waveform used in polygonal zone searching.	5-116
:VPOSition	Sets or queries the vertical position used in polygonal zone searching.	5-117
:ZNUMber	Sets or queries the zone number used in polygonal zone searching.	5-117
:NUMBer<x>		
:RECTangle?	Queries all waveform rectangular zone search settings.	5-117
:RECTangle		
:HORizontal	Sets or queries the horizontal position of the rectangle used in rectangular zone searching.	5-117
:RANGe	Sets or queries the source window of the rectangle used in rectangular zone searching.	5-117
:TRACe	Sets or queries the source waveform of the rectangle used in rectangular zone searching.	5-117
:VERTical	Sets or queries the vertical position of the rectangle used in rectangular zone searching.	5-118
:NUMBer<x>		
:WAVE?	Queries all waveform waveform zone search settings.	5-118
:WAVE		
:RANGe	Sets or queries the source window used in waveform zone searching.	5-118
:TRACe	Sets or queries the source waveform used in waveform zone searching.	5-118
:TRANge	Sets or queries the determination period used in waveform zone searching.	5-118
:ZNUMber	Sets or queries the zone number used in waveform zone searching.	5-118
:SIMPle?	Queries all simple searching settings.	5-118
:SIMPle		
:HORizontal	Sets or queries the horizontal position of the rectangle to use in simple searching.	5-118
:RANGe	Sets or queries the target window of the rectangle to use in simple searching.	5-119
:TRACe	Sets or queries the source trace of the rectangle to use in simple searching.	5-119
:VERTical	Sets or queries the vertical position of the rectangle to use in simple searching.	5-119
:TIME?	Queries the time of the source record number.	5-119

IMAGe Group

:IMAGe?	Queries all screen capture data output settings.	5-120
:IMAGe		
:ABORt	Aborts saving screen capture data to the storage device.	5-120
:BACKground	Sets or queries the screen capture background.	5-120
:EXECute	Saves the screen capture data to the storage device.	5-120
:FORMat	Sets or queries the screen capture output format.	5-120
:INFormation	Sets or queries whether setting information is included in screen capture data.	5-120
:MODE	Sets or queries the screen capture output mode.	5-120
:SAVE?	Queries all file output settings.	5-120
:SAVE		
:ANAMing	Sets or queries the on/off state of the auto naming feature for saving files.	5-120
:CDIRectory	Changes the file directory.	5-120
:DRIVe	Sets the storage device that the file will be created in.	5-120
:NAME	Sets or queries the name of the file to be created.	5-121
:SEND?	Queries the screen capture data value.	5-121
:TONE	Sets or queries the color tone of the screen capture data that will be saved.	5-121

INITialize Group

:INITialize		
:EXECute	Executes initialization.	5-122
:UNDO	Undoes the initialization that was executed.	5-122

LOGic Group

:LOGic?	Queries all logic input waveform settings.	5-123
:LOGic		
:MODE	Sets or queries the logic input on/off state.	5-123
[:PODA]?	Queries all logic input settings.	5-123

Command	Function	Page
[:PODA]		
:BITOrder	Sets or queries bit order of the logic input.	5-123
:BITSetup		
:ALL		
DISPlay	Turns on or off all bit displays of the specified pod of the logic input.	5-123
:LEVel	Sets or queries the user-defined threshold level of the specified pod of the logic input.	5-123
:TYPE	Selects the threshold level of the specified pod of the logic input.	5-123
:BIT<x>?	Queries all settings of a bit of the specified pod of the logic input.	5-123
:BIT		
:DISPlay	Turns on or off all bit displays of the specified pod of the logic input.	5-123
:LABel	Sets or queries the label of a bit of the specified pod of the logic input.	5-124
:LEVel	Sets or queries the user-defined threshold level of a bit of the specified pod of the logic input.	5-124
:TYPE	Selects the threshold level of a bit of the specified pod of the logic input.	5-124
:BUS?	Queries all logic input bus settings.	5-124
:BUS		
:ASSignment	Sets or queries the logic input bus assignment.	5-124
:DISPlay	Sets or queries the on/off state of the logic input bus display.	5-124
:FORMat	Sets or queries the logic input bus display format.	5-124
:LABel	Sets or queries the label of the logic input bus label.	5-124
:DESKew	Sets or queries the deskewing of the specified pod of the logic input.	5-124
:HYSTeresis	Sets or queries the hysteresis of the specified pod of the logic input.	5-124
:STATe		
:ASSignment?	Queries the logic input state display assignment.	5-124
:ASSignment		
:ALL	Sets or queries all settings related to the logic input state display assignment.	5-125
:BIT	Sets or queries the logic input state display assignments of all bits.	5-125
:BUS	Sets or queries the bus setting of the logic input state display assignment.	5-125
:CLOCk	Sets or queries the state display reference clock waveform of the logic input.	5-125
:HYSTeresis	Sets or queries the hysteresis of the state display reference clock waveform of the logic input.	5-125
:MODE	Sets or queries the on/off state of the logic input state display.	5-125
:POLarity	Sets or queries the polarity of the state display reference clock waveform of the logic input.	5-125
:THReshold	Sets or queries the detection level of the state display reference clock waveform of the logic input.	5-125
:POSition	Sets or queries the vertical position of the logic signal.	5-125
:SIZE	Sets or queries the display size of the logic signal.	5-126

MATH Group

:MATH<x>?	Queries all computation settings.	5-127
:MATH<x>		
:DISPlay	Sets or queries whether to show (ON) or hide (OFF) the computed waveforms.	5-127
:ECOunt?	Queries all edge count settings.	5-127
:ECOunt		
:HYSTeresis	Sets or queries the edge detection level hysteresis for edge count.	5-127
:POLarity	Sets or queries the edge detection polarity for edge count.	5-127
:THReshold	Sets or queries the edge detection Level for edge count computation.	5-127
:FILTer?	Queries all filter settings.	5-127
:FILTer		
:FORDer	Sets or queries the IIR filter's filter order.	5-127
:HCUToff	Sets or queries the cutoff frequency of the IIR filter high-pass filter.	5-127
:LCUToff	Sets or queries the cutoff frequency of the IIR filter low-pass filter.	5-128
:TIME	Sets or queries the phase shift delay.	5-128
:TYPE	Sets or queries the filter type.	5-128
:WEIGht	Sets or queries the moving average weight.	5-128
:INTegral?	Queries all integral settings.	5-128
:INTegral		

Command	Function	Page
:SPOint	Sets or queries the integral start position.	5-128
:LABel?	Queries all settings related to the label of the computed waveform.	5-128
:LABel		
[:DEFine]	Sets or queries a computed waveform label.	5-128
:MODE	Sets or queries the label display on/off state of computed waveforms.	5-128
:OPERation	Sets or queries the operator.	5-129
:RCOunt?	Queries all rotary count computation settings.	5-129
:RCOunt		
HYSTeresis<y>	Sets or queries the hysteresis for rotary count computation.	5-129
:THReshold<y>	Sets or queries the determination level for rotary count computation.	5-129
ZPOLarity	Sets or queries the polarity for rotary count computation.	5-129
:SCALe?	Queries all scaling settings.	5-129
:SCALe		
:CENTer	Sets or queries the center value for manual scaling.	5-129
:MODE	Sets or queries the scale mode.	5-129
:SENSitivity	Sets or queries the sensitivity for manual scaling.	5-130
:UNIT?	Queries all computation unit settings.	5-130
:UNIT		
[:DEFine]	Sets or queries the computation unit.	5-130
:MODE	Sets or queries whether a computation unit will be attached automatically or manually.	5-130
:USERdefine?	Queries all user-defined computation settings.	5-130
:USERdefine		
:AVERage?	Queries all settings related to the averaging of user-defined computation.	5-130
:AVERage		
:EWEight	Sets or queries the attenuation constant of exponential averaging of user-defined computation.	5-130
:MODE	Sets or queries the averaging mode for user-defined computation.	5-130
:USERdefine		
:CONSitant<y>	Sets or queries a constant for user-defined computation.	5-130
:DEFine	Sets or queries an expression for user-defined computation.	5-130
:FILTer<y>?	Queries all settings related to the filter of user-defined computation.	5-131
:FILTer<y>		
:BAND	Sets or queries the filter band (frequency band) for user-defined computation.	5-131
:CUToff<z>	Sets or queries an cutoff frequency for user-defined computation.	5-131
:TYPE	Sets or queries the filter type for user-defined computation.	5-131
:USERdefine		
:HISTory		
:ABORt	Aborts the user-defined computation of history waveforms (Math on History).	5-131
:EXECute	Executes the user-defined computation of history waveforms (Math on History).	5-131
:USERdefine		
:SCALe?	Queries all settings related to the scaling of user-defined computation.	5-131
:SCALe		
:ARANging	Executes auto range on user-defined computation.	5-131
:CENTer	Sets or queries the center value of scaling for user-defined computation.	5-131
:SENSitivity	Sets or queries the scaling sensitivity for user-defined computation.	5-131

MEASure Group

:MEASure?	Queries all the settings for automated measurement of waveform parameters.	5-132
:MEASure		
:BIT<x>?	Queries all the parameter on/off states of the specified logic waveform.	5-132
:BIT<x>		
:ALL	Collectively turns on or off all measurement items of a logic waveform.	5-132
:AREA2?	Queries all the parameter on/off states of a logic waveform in Area2.	5-132
:AREA2		
:ALL	Collectively turns on or off all measurement items of a logic waveform in Area2.	5-132
:<Parameter>?	Queries a waveform parameter setting of a logic waveform in Area2.	5-132
:<Parameter>		

Command	Function	Page
:COUNt?	Queries the statistical processing count of a logic waveform parameter in Area2.	5-132
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a logic waveform parameter in Area2.	5-132
:STATe	Sets or queries the on/of state of a logic waveform parameter in Area2.	5-132
:VALue?	Queries automated measurement values of waveform parameters of a logic waveform in Area2.	5-133
:AREA2		
:COPY	Copies the on/off state of all measurement items of a logic waveform to all other waveforms in Area2.	5-133
:DELay?	Queries all settings related to the delay between channels of a logic waveform in Area2.	5-133
:DELay		
:MEASure?	Queries all of the source waveform settings for measuring the delay between channels of a logic waveform in Area2.	5-133
:MEASure		
:COUNt	Sets or queries the count number of a source waveform for measuring the delay between channels of a logic waveform in Area2.	5-133
:SLOPe	Sets or queries the slope of a source waveform for measuring the delay between channels of a logic waveform in Area2.	5-133
:DELay		
:REFerence?	Queries all of the reference waveform settings for measuring the delay between channels of a logic waveform in Area2.	5-133
:REFerence		
:COUNt	Sets or queries the count number of the reference waveform for measuring the delay between channels of a logic waveform in Area2.	5-134
:SLOPe	Sets or queries the slope of the reference waveform for measuring the delay between channels of a logic waveform in Area2.	5-134
:SOURce	Sets or queries whether the reference point for measuring the delay between channels of a logic waveform will be set to the trigger point or a waveform in Area2.	5-134
:TRACe	Sets or queries the edge of the reference waveform for measuring the delay between channels of a logic waveform in Area2.	5-134
:DELay		
:STATe	Sets or queries the on/off state and display format of the delay parameter in Area2.	5-134
:BIT<x>		
:<Parameter>?	Queries a waveform parameter setting of a logic waveform.	5-134
:<Parameter>		
:COUNt?	Queries the statistical processing count of a logic waveform parameter.	5-134
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a logic waveform parameter.	5-135
:STATe	Sets or queries the on/of state of a logic waveform parameter.	5-135
:VALue?	Queries automated measurement values of waveform parameters of a logic waveform.	5-135
:BIT<x>		
:COPY	Copies the on/off state of all measurement items of a logic waveform to all other waveforms.	5-135
:DELay?	Queries all settings related to the delay between channels of a logic waveform.	5-135
:DELay		
:MEASure?	Queries all the settings for a source waveform for measuring the delay between channels of a logic waveform.	5-135
:MEASure		
:COUNt	Sets or queries the source waveform count for measuring the delay between channels of a logic waveform.	5-136
:SLOPe	Sets or queries a source waveform slope that will be used to measure delay between channels for a logic waveform.	5-136
:DELay		
:REFerence?	Queries all the settings for the reference waveform for measuring the delay between channels of a logic waveform.	5-136
:REFerence		
:COUNt	Sets or queries the reference waveform count for measuring the delay between channels of a logic waveform.	5-136

Command	Function	Page
:SLOPe	Sets or queries the reference waveform slope that will be used to measure delay between channels for a logic waveform.	5-136
:SOURce	Sets or queries whether the reference point for measuring the delay between channels of a logic waveform will be set to the trigger point or a logic waveform.	5-136
:TRACe	Sets or queries the reference waveform edge that will be used to measure delay between channels for a logic waveform.	5-136
:DELay		
:STATe	Sets or queries the on/off state and display format of the delay parameter.	5-137
:{CHANnel<x>|MATH<x>}?	Queries all the parameter on/off states of a waveform.	5-137
:{CHANnel<x>|MATH<x>}		
:ALL	Collectively turns on or off all measurement items of a waveform.	5-137
:AREA2?	Queries all the parameter on/off states of a waveform in Area2.	5-137
:AREA2		
:ALL	Collectively turns on or off all measurement items of a waveform in Area2.	5-137
:<Parameter>?	Queries a waveform parameter setting of a waveform in Area2.	5-137
:<Parameter>		
:COUNt?	Queries the statistical processing count of a waveform parameter in Area2.	5-137
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a waveform parameter in Area2.	5-138
:STATe	Sets or queries the on/of state of a waveform parameter in Area2.	5-138
:VALue?	Queries the automated measurement value of a waveform parameter in Area2.	5-138
:AREA2		
:COPY	Copies the on/off state of all measurement items of a waveform to all other waveforms in Area2.	5-138
:CYCLe	Sets or queries cycle mode of Area2.	5-138
:DELay?	Queries all settings related to the delay between channels of a waveform in Area2.	5-139
:DELay		
MEASure?	Queries all of the source waveform settings for measuring the delay between channels of a waveform in Area2.	5-139
:MEASure		
:COUNt	Sets or queries the count number of a source waveform for measuring the delay between channels of a waveform in Area2.	5-139
:SLOPe	Sets or queries the slope of a source waveform for measuring the delay between channels of a waveform in Area2.	5-139
:DELay		
REFerence?	Queries all of the reference waveform settings for measuring the delay between channels of a waveform in Area2.	5-139
:REFerence		
:COUNt	Sets or queries the count number of the reference waveform for measuring the delay between channels of a waveform in Area2.	5-139
:SLOPe	Sets or queries the slope of the reference waveform for measuring the delay between channels of a waveform in Area2.	5-139
:SOURce	Sets or queries whether the reference point for measuring the delay between channels of a waveform will be set to the trigger point or a waveform in Area2.	5-140
:TRACe	Sets or queries the edge of the reference waveform for measuring the delay between channels of a waveform in Area2.	5-140
:DELay		
:STATe	Sets or queries the on/off state and display format of the delay parameter in Area2.	5-140
:{CHANnel<x>|MATH<x>}		
:<Parameter>	Queries a waveform parameter setting of a waveform.	5-140
:<Parameter>		
:COUNt?	Queries the statistical processing count of a waveform parameter.	5-140
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a waveform parameter.	5-140
:STATe	Sets or queries the on/of state of a waveform parameter.	5-141
:VALue?	Queries automated measurement values of waveform parameters of a waveform.	5-141
:{CHANnel<x>|MATH<x>}		

Command	Function	Page
:COPY	Copies the on/off state of all measurement items of a waveform to all other waveforms.	5-141
:CYCLe	Sets or queries the cycle mode.	5-142
:DELay?	Queries all settings related to the delay between channels of a waveform.	5-142
:DELay		
:MEASure?	Queries all the settings for a source waveform for measuring the delay between channels of a waveform.	5-142
:MEASure		
:COUNt	Sets or queries the number of edges at which delay between channels will be measured for a waveform.	5-142
:SLOPe	Sets or queries a source waveform slope that will be used to measure delay between channels for a waveform.	5-142
:DELay		
:REFerence?	Queries all reference waveform settings used to measure the delay between channels for the specified waveform.	5-142
:REFerence		
:COUNt	Sets or queries the reference waveform count for measuring the delay between channels of a waveform.	5-142
:SLOPe	Sets or queries the reference waveform slope that will be used to measure delay between channels for a waveform.	5-143
:SOURce	Sets or queries whether to set the reference point for measuring the delay between channels for a waveform to a trigger point or to a waveform.	5-143
:TRACe	Sets or queries the reference waveform edge that will be used to measure delay between channels for a waveform.	5-143
:DELay		
:STATe	Sets or queries the on/off state and display format of the delay parameter.	5-143
:DPRoximal?	Queries all distal, mesial, and proximal settings.	5-143
:DPRoximal		
:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-143
:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-143
:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-144
:{CHANnel<x>|MATH<x>}		
:METHod	Sets or queries the high and low points.	5-144
:CONTinuous?	Queries all settings for the normal statistical processing of automatically measured waveform parameters.	5-144
:CONTinuous		
:RESTart	Restarts the normal statistical processing of automatically measured waveform parameters.	5-144
:TLCHange	Sets or queries whether normal statistical processing of automatically measured waveform parameters is restarted when the trigger level is changed.	5-144
:CYCLe?	Queries all settings related to the cyclic statistical processing of automated measurement.	5-144
:CYCLe		
:ABORt	Aborts the execution of the cyclic statistical processing of automated measurement.	5-144
:EXECute	Executes the cyclic statistical processing of automated measurement.	5-144
:TRACe	Sets or queries the source waveform of the cyclic statistical processing of automated measurement.	5-144
:HISTory		
:ABORt	Aborts the execution of the statistical processing of the history waveform of automated measurement.	5-144
:EXECute	Executes the statistical processing of the history waveform of automated measurement.	5-144
:INDicator	Sets or queries the measurement location indicator.	5-145
:MODE	Sets or queries the automated measurement mode.	5-145
:RANGe<x>	Sets or queries the measurement source window.	5-145
:TRANge<x>	Sets or queries the measurement time period.	5-145
:USER<x>?	Queries all automatic measurement settings for a Calc item.	5-145
:USER<x>		
:COUNt?	Queries the statistical processing count of the automatically measured value of a Calc item.	5-145
:DEFine	Sets or queries the expression for the automatically measured value of a Calc item.	5-145

Command	Function	Page
:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value that is calculated on the automatically measured value of a Calc item.	5-145
:NAME	Sets or queries the name of a Calc item.	5-146
:STATe	Sets or queries the on/off status of automated measurement of a Calc item.	5-146
:UNIT	Sets or queries the unit of a Calc item.	5-146
:VALue?	Queries the automatically measured value of a Calc item.	5-146
:WAIT?	Waits for the completion of automated measurement with a timeout.	5-146
:ZLINkage	Sets or queries the linkage (on/off) between the measurement numbers and zoom positions of the results of automated measurement of waveform parameters.	5-146

RECall Group

:RECall		
:SETup<x>		
:EXECute	Recalls setup data from the internal memory.	5-147

REFerence Group

:REFerence<x>?	Queries all reference waveform settings.	5-148
:REFerence<x>		
:DISPlay	Sets or queries the reference waveform display on/off state.	5-148
:LABel?	Queries all settings related to the label of a reference.	5-148
:LABel		
[:DEFine]	Sets or queries the label of a reference.	5-148
:MODE	Sets or queries the label display on/off state of a reference.	5-148
:LOAD	Loads a reference waveform.	5-148
:POSition	Sets or queries the vertical position of the reference waveform.	5-148

SEARch Group

:SEARch?	Queries all waveform search settings.	5-149
:SEARch		
:ABORt	Aborts the search.	5-149
:ASCRoll<x>?	Queries all auto scroll settings.	5-149
:ASCRoll<x>		
:JUMP	Moves the center position of the zoom box to the left or right edge of the main window.	5-149
:SPEed	Sets or queries the auto scroll speed of the zoom box.	5-149
:STARt	Starts auto scrolling.	5-149
:STOP	Stops auto scrolling.	5-149
:EDGE?	Queries all edge search settings.	5-149
:EDGE		
:HYSTeresis	Sets or queries the edge search level hysteresis.	5-149
:LEVel	Sets or queries the edge search level.	5-149
:SLOPe	Sets or queries the edge search slope.	5-149
:SOURce	Sets or queries the source waveform of edge search.	5-150
:EPOint	Sets or queries the search end position.	5-150
:EXECute	Executes a search. This is an overlap command.	5-150
:MARK	Sets or queries the detected point mark on/off state.	5-150
:MAG<x>	Sets or queries the zoom factor of the zoom window.	5-150
:POSition<x>	Sets or queries the position of a zoom box.	5-150
:PATTern?	Queries all settings related to the combination (pattern) search of multiple inputs.	5-150
:PATTern		
:BITS?	Queries all settings related to the bits of the combination (pattern) search of multiple inputs.	5-150
:BITS		
:HEXa	Sets the bit pattern for the combination (pattern) search of multiple inputs in hexadecimal notation.	5-150
:PATTern	Sets or queries the bit pattern for the combination (pattern) search of multiple inputs in binary notation.	5-150
:{CHANnel<x>|MATH<x>}?	Queries all settings related to a waveform of the combination (pattern) search of multiple inputs.	5-150
:{CHANnel<x>|MATH<x>}		

Command	Function	Page
:HYSTeresis	Sets or queries the hysteresis of the source waveform for the combination (pattern) search of multiple inputs.	5-151
:LEVel	Sets or queries the search level of the source waveform for the combination (pattern) search of multiple inputs.	5-151
:PATTern	Sets or queries the search pattern of the source waveform for the combination (pattern) search of multiple inputs.	5-151
:CLOCk?	Queries all settings related to the clock channel of the combination (pattern) search of multiple inputs.	5-151
:CLOCk		
:HYSTeresis	Sets or queries the hysteresis of the clock channel for the combination (pattern) search of multiple inputs.	5-151
:LEVel	Sets or queries the search level of the clock channel for the combination (pattern) search of multiple inputs.	5-151
:SLOPe	Sets or queries the slope of the clock channel for the combination (pattern) search of multiple inputs.	5-151
:SOURce	Sets or queries the source waveform of the clock channel for the combination (pattern) search of multiple inputs.	5-152
:CONDition	Sets or queries the achievement condition of the source waveform for the combination (pattern) search of multiple inputs.	5-152
:LOGic	Sets or queries the logic of the source waveform for the combination (pattern) search of multiple inputs.	5-152
:TIME<x>	Sets or queries the reference time of the source waveform for the combination (pattern) search of multiple inputs.	5-152
:TQUalify	Sets or queries the time condition of the source waveform for the combination (pattern) of multiple inputs.	5-152
:WIDTh?	Queries all pulse width search settings.	5-152
:PULSe		
:HYSTeresis	Sets or queries the pulse width search level hysteresis.	5-152
:LEVel	Sets or queries the pulse width search level.	5-152
:POLarity	Sets or queries the pulse width search level polarity.	5-152
:SOURce	Sets or queries the source waveform of pulse width search.	5-152
:TIME<x>	Sets or queries the reference time of pulse width search.	5-153
:TQUalify	Sets or queries the time condition for pulse width searching.	5-153
:SELect	Queries the setting of the detected point to show in the zoom window and the zoom position of that point.	5-153
:SELect?		
MAXimum	Queries the total number of detected points.	5-153
:SKIP?	Queries all skip mode settings.	5-153
:SKIP		
:DECimation	Sets or queries the decimation detection of skip mode.	5-153
:HOLDoff	Sets or queries the hold-off detection.	5-153
:MODE	Sets or queries the mode of skip mode.	5-153
:SPOint	Sets or queries the search start position.	5-153
:TIMeout?	Queries all timeout period search settings.	5-153
:TIMeout		
:POLarity	Sets or queries the timeout period search polarity.	5-153
:SOURce	Sets or queries the timeout period search source waveform.	5-154
:TIME	Sets or queries the timeout period of the timeout period search.	5-154
:TWINdow	Sets or queries the zoom window for showing the detected area.	5-154
:TYPE	Sets or queries the search type.	5-154

SERialbus Group

:SERialbus<x>?	Queries all serial bus signal analysis and search settings.	5-155
:SERialbus<x>		
:ASETup		
:ABORt	Aborts the serial bus signal auto setup.	5-155
:EXECute	Execute an serial bus signal auto setup.	5-155
:CAN?	Queries all CAN bus signal settings.	5-155
:CAN		
:ANALyze?	Queries all CAN bus signal analysis settings.	5-155
[:ANALyze]		
:SETup?	Queries all CAN bus signal analysis bus settings.	5-155
:SETup		

Command	Function	Page
:BRATe	Sets or queries the CAN bus signal analysis bit rate (data transfer rate).	5-155
:RECessive	Sets or queries the CAN bus signal analysis recessive level.	5-156
:SOURce	Sets or queries the CAN bus signal analysis source.	5-156
:SPOint	Sets or queries the CAN bus signal analysis sample point.	5-156
:DETail?	Queries all CAN bus signal analysis result list settings.	5-156
:DETail		
:DISPlay	Sets or queries the display mode for the CAN bus signal analysis result list.	5-156
:LIST		
:ITEM?	Queries all items that will be displayed in the CAN bus signal analysis result list.	5-156
:VALue?	Queries all the data for the specified analysis number in the CAN bus signal analysis result list.	5-156
:SEARch?	Queries all CAN bus signal search settings.	5-156
:SEARch		
:ABORt	Aborts the CAN bus signal search.	5-156
:EXECute	Executes a CAN bus signal search.	5-157
:FJUMp		
:ACK	Jumps to the ACK field in the CAN bus signal search result.	5-157
:CONTrol	Jumps to the control field in the CAN bus signal search result.	5-157
:CRC	Jumps to the CRC field in the CAN bus signal search result.	5-157
:DATA	Jumps to the data field in the CAN bus signal search result.	5-157
:IDENtifier	Jumps to the identifier in the CAN bus signal search result.	5-157
:SOF	Jumps to the SOF in the CAN bus signal search result.	5-157
:SELect	Sets which detected point to display in the CAN bus signal search zoom window and queries the zoom position of the detected point.	5-157
:SELect?		
MAXimum	Queries the number of detected points in the CAN bus signal search.	5-157
:SETup?	Queries all CAN bus signal search condition settings.	5-157
:SETup		
:EFRame?	Queries all CAN bus signal search error settings.	5-157
:EFRame		
[:MODE]	Sets or queries the CAN bus signal search error frame setting.	5-158
:CRC	Sets or queries the CAN bus signal search CRC error setting.	5-158
:STUFF	Sets or queries the CAN bus signal search stuff error setting.	5-158
:IDData?	Queries all ID and data condition settings for CAN bus signal searching.	5-158
[:IDData]		
:ACK?	Queries all ID and data condition ACK settings for CAN bus signal searching.	5-158
:ACK		
:MODE	Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN bus signal searching.	5-158
:TYPE	Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-158
:DATA?	Queries all data settings for the ID and data conditions for CAN bus signal searching.	5-158
:DATA		
:CONDition	Sets or queries the data comparison condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-159
:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal searching in decimal notation.	5-159
:DLC	Sets or queries the number of significant bytes (DLC), which is one of the ID and data conditions, for CAN bus signal searching.	5-159
:DPATtern		
:HEXa	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal searching in hexadecimal notation.	5-159
:PATTern	Sets or queries the data, which is one of the ID and data conditions, for CAN bus signal searching in binary notation.	5-159
:ENDian	Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN bus signal searching.	5-159
:MODE	Sets or queries the data condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal searching.	5-160
:MSBLsb	Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN bus signal searching.	5-160

Command	Function	Page
:SIGN	Sets or queries the sign, which is one of the ID and data conditions, for CAN bus signal searching.	5-160
:IDENtifier?	Queries the ID, which is one of the ID and data conditions, for CAN bus signal searching.	5-160
:IDENtifier		
:HEXa	Sets an ID, which is one of the ID and data conditions, for CAN bus signal searching in hexadecimal notation.	5-160
:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN bus signal searching.	5-160
:MODE	Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal searching.	5-160
:PATTern	Sets or queries the ID condition, which is one of the ID and data conditions, for CAN bus signal searching in binary notation.	5-161
:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CAN bus signal searching.	5-161
:MSIGnal?	Queries the message signal, which is one of the ID and data conditions, for CAN bus signal searching.	5-161
:MSIGnal		
:MESSage		
:ITEM	Sets the message item, which is one of the ID and data conditions, for CAN bus signal searching.	5-161
:SELect	Sets or queries the message signal condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-161
:SIGNal?	Queries the signal, which is one of the ID and data conditions, for CAN bus signal searching.	5-161
:SIGNal		
:CONDition	Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-161
:DECimal<y>	Sets a reference signal value, which is one of the ID and data conditions, for CAN bus signal searching in decimal notation.	5-162
:ITEM	Sets the signal item, which is one of the ID and data conditions, for CAN bus signal searching.	5-162
:RTR	Sets or queries the RTR value, which is one of the ID and data conditions, for CAN bus signal searching.	5-162
:MODE	Sets or queries the CAN bus signal search mode setting.	5-162
:CANFD?	Queries all CAN FD bus signal settings.	5-162
:CANFD		
:ANALyze?	Queries all CAN FD bus signal analysis settings.	5-162
[:ANALyze]		
:SETup?	Queries all CAN FD bus signal analysis bus settings.	5-162
:SETup		
:BRATe	Sets or queries the CAN FD bus signal analysis bit rate (data transfer rate).	5-162
:DBRate	Sets or queries the CAN FD bus signal analysis data bit rate (data transfer rate of data phase).	5-163
:DSPoint	Sets or queries the CAN FD bus signal analysis data phase sample point.	5-163
:FDSTandard	Sets or queries whether the CAN FD bus signal to be analyzed is an ISO standard signal.	5-163
:RECessive	Sets or queries the CAN FD bus signal analysis recessive level.	5-163
:SOURce	Sets or queries the CAN FD bus signal analysis source.	5-163
:SPOint	Sets or queries the CAN FD bus signal analysis sample point.	5-163
:DETail?	Queries all CAN FD bus signal analysis result list settings.	5-164
:DETail		
:DISPlay	Sets or queries the display mode for the CAN FD bus signal analysis result list.	5-164
:LIST		
:ITEM?	Queries all items that will be displayed in the CAN FD bus signal analysis result list.	5-164
:VALue?	Queries all the data for the specified analysis number in the CAN FD bus signal analysis result list.	5-164
:SEARch?	Queries all CAN FD bus signal search settings.	5-164
:SEARch		
:ABORt	Aborts the CAN FD bus signal search.	5-164
:EXECute	Executes a CAN FD bus signal search.	5-164
:FJUMp		

Command	Function	Page
:ACK	Jumps to the ACKField in the CAN FD bus signal search result.	5-164
:CONTrol	Jumps to the control field in the CAN FD bus signal search result.	5-164
:CRC	Jumps to the CRC field in the CAN FD bus signal search result.	5-164
:DATA	Jumps to the data field in the CAN FD bus signal search result.	5-164
:IDENtifier	Jumps to the identifier in the CAN FD bus signal search result.	5-164
:SOF	Jumps to the SOF in the CAN FD bus signal search result.	5-164
:SELect	Sets which detected point to display in the CAN FD bus signal search zoom window and queries the zoom position of the detected point.	5-165
:SELect?		
MAXimum	Queries the number of detected points in the CAN FD bus signal search.	5-165
:SETup?	Queries all CAN FD bus signal search condition settings.	5-165
:SETup		
:EFRame?	Queries all CAN FD bus signal search error settings.	5-165
:EFRame		
:CRC	Sets or queries the CAN FD bus signal search CRC error setting.	5-165
:CRCEFactor?	Queries all CRC error factor settings for CAN FD bus signal searching.	5-165
:CRCEFactor		
:CRCSequence	Sets or queries the CRC sequence (a CRC error factor) for CAN FD bus signal searching.	5-165
:SCOunt	Sets or queries the Stuff Count (a CRC error factor) for CAN FD bus signal searching.	5-165
:FSTuff	Sets or queries the CAN FD bus signal search fixed stuff error setting.	5-166
[:MODE]	Sets or queries the CAN FD bus signal search error frame setting.	5-166
:STUFF	Sets or queries the CAN FD bus signal search stuff error setting.	5-166
:FDF		
:CONDition	Sets or queries the CAN FD bus signal search FDF condition setting.	5-166
:IDData?	Queries all ID and data condition settings for CAN FD bus signal searching.	5-166
[:IDData]		
:ACK?	Queries all ACK settings for the ID and data conditions for CAN FD bus signal searching.	5-166
:ACK		
:MODE	Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-166
:TYPE	Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:DATA?	Queries all data settings for the ID and data conditions for CAN FD bus signal searching.	5-167
:DATA		
:BCOunt	Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:CONDition	Sets or queries the data comparison condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in decimal notation.	5-167
:DPATtern		
:HEXa	Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in hexadecimal notation.	5-168
:PATTern	Sets or queries the data value, which is one of the ID and data conditions, for CAN FD bus signal searching in binary notation.	5-168
:ENDian	Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-168
:MODE	Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-168
:MSBLsb	Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-168
:SIGN	Sets or queries whether signed or unsigned data format will be used, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-168
:IDENtifier?	Queries all identifier settings for the ID and data conditions for CAN FD bus signal searching.	5-169
:IDENtifier		

Command	Function	Page
:HEXa	Sets the ID value, which is one of the ID and data conditions, for CAN FD bus signal searching in hexadecimal notation.	5-169
:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN FD bus signal searching.	5-169
:MODE	Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-169
:PATTern	Sets or queries the ID pattern, which is one of the ID and data conditions, for CAN FD bus signal searching in binary notation.	5-169
:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-169
:MSIGnal?	Queries all message and signal settings for the ID and data conditions for CAN FD bus signal searching.	5-169
:MSIGnal		
:MESSage		
:ITEM	Sets the message item, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-169
:SELect	Sets or queries the message and signal condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-170
:SIGNal?	Queries all signal settings for the ID and data conditions for CAN FD bus signal searching.	5-170
:SIGNal		
:CONDition	Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-170
:SIGNal		
:DECimal<y>	Sets a signal's reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in decimal notation.	5-170
:ITEM	Sets the signal item, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-170
:RTR	Sets or queries the RTR value, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-170
:MODE	Sets or queries the CAN FD bus signal search mode setting.	5-171
:CXPI?	Queries all CXPI bus signal analysis and search settings.	5-171
:CXPI		
:ANALyze?	Queries all CXPI bus signal analysis settings.	5-171
[:ANALyze]		
:SETup?	Queries all CXPI bus signal analysis bus settings.	5-171
:SETup		
:BRATe	Sets or queries the CXPI bus signal analysis bit rate (data transfer rate).	5-171
:CEDetection	Sets or queries the enable/disable condition of counter error detection for CXPI bus signal analysis.	5-171
:CTOLerance	Sets or queries the CXPI bus signal analysis clock tolerance.	5-171
:SOURce	Sets or queries the CXPI bus signal analysis source.	5-171
:TSAMple	Sets or queries the logic value (1 or 0) determination threshold for CXPI bus signal analysis.	5-172
:DETail?	Queries all CXPI bus signal analysis result list settings.	5-172
:DETail		
:DISPlay	Sets or queries the display mode for the CXPI bus signal analysis result list.	5-172
:LIST		
:ALL?	Queries all the data for all analysis numbers in the CXPI bus signal analysis result list.	5-172
:ITEM?	Queries all items that will be displayed in the CXPI bus signal analysis result list.	5-172
:VALue?	Queries all the data for the specified analysis number in the CXPI bus signal analysis result list.	5-172
:SEARch?	Queries all CXPI bus signal search settings.	5-172
:SEARch		
:ABORt	Aborts the CXPI bus signal search.	5-172
:EXECute	Executes a CXPI bus signal search.	5-172
:SELect	Sets which detected point to display in the CXPI bus signal search zoom window and queries the zoom position of the detected point.	5-172
:SETup?	Queries all CXPI bus signal search condition settings.	5-173
:SETup		
:ERRor?	Queries all CXPI bus signal search error settings.	5-173

Command	Function	Page
:ERRor		
:CLOCk	Sets or queries the CXPI bus signal search clock error setting.	5-173
:COUNter	Sets or queries the CXPI bus signal search counter error setting.	5-173
:CRC	Sets or queries the CXPI bus signal search CRC error setting.	5-173
:DLENgth	Sets or queries the CXPI bus signal search data length error setting.	5-173
:FRAMing	Sets or queries the CXPI bus signal search framing error setting.	5-173
:IBS	Sets or queries the CXPI bus signal search IBS error setting.	5-174
:PARity	Sets or queries the CXPI bus signal search parity error setting.	5-174
:IDData?	Queries all ID and data condition settings for CXPI bus signal searching.	5-174
[:IDData]		
:DATA?	Queries all data settings for the ID and data conditions for CXPI bus signal searching.	5-174
:DATA		
:BCOunt	Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CXPI bus signal searching.	5-174
:CONDition	Sets or queries the data comparison condition, which is one of the ID and data conditions, for CXPI bus signal searching.	5-174
:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for CXPI bus signal searching.	5-174
:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in decimal notation.	5-175
:DPATtern		
:HEXa	Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in hexadecimal notation.	5-175
:DPATtern		
:PATTern	Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in binary notation.	5-175
:ENDian	Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CXPI bus signal searching.	5-175
:MODE	Sets or queries the ID and data condition (enabled/disabled) for CXPI bus signal searching.	5-175
:MSBLsb	Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CXPI bus signal searching.	5-175
:SIGN	Sets or queries the sign, which is one of the ID and data conditions, for CXPI bus signal searching.	5-176
:FINFormation?	Queries all frame information settings for the ID and data conditions for CXPI bus signal searching.	5-176
:FINFormation		
:CT	Sets or queries the frame information counter value, which is one of the ID and data conditions, for CXPI bus signal searching.	5-176
:MODE	Sets or queries the frame information (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal searching.	5-176
:SLEEP	Sets or queries the frame information sleep bit, which is one of the ID and data conditions, for CXPI bus signal searching.	5-176
:WAKeup	Sets or queries the frame information wakeup bit, which is one of the ID and data conditions, for CXPI bus signal searching.	5-176
:ID?	Queries all ID settings for the ID and data conditions for CXPI bus signal searching.	5-177
:ID		
:HEXa	Sets an ID, which is one of the ID and data conditions, for CXPI bus signal searching in hexadecimal notation.	5-177
:MODE	Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal searching.	5-177
:PATTern	Sets or queries the ID, which is one of the ID and data conditions, for CXPI bus signal searching in binary notation.	5-177
:PTYPE	Sets or queries the PTYPE comparison condition, which is one of the ID and data conditions, for CXPI bus signal searching.	5-177
:MODE	Sets or queries the CXPI bus signal search type.	5-177
:PTYPE?	Queries all CXPI bus signal search PTYPE settings.	5-177
:PTYPE		
:CONDition	Sets or queries the CXPI bus signal search PTYPE condition setting.	5-177
:WAKeupsleep?	Queries all wakeup and sleep settings for CXPI bus signal searching.	5-178
:WAKeupsleep		
:SFRame	Sets or queries the CXPI bus signal search sleep frame setting.	5-178

Command	Function	Page
:SLEEP	Sets or queries the CXPI bus signal search sleep (clock unavailable condition) setting.	5-178
:WAKeup	Sets or queries the CXPI bus signal search wakeup (clock available condition) setting.	5-178
:WPULse	Sets or queries the CXPI bus signal search wakeup pulse setting.	5-178
:DECode?	Queries all settings related to the decoding of serial bus signals.	5-178
:DECode		
[:FORMat]	Sets or queries the decode display format of serial bus signals.	5-178
:SSCMode	Sets or queries the decode display of the start/stop condition of I2C serial bus signals.	5-178
:DISPlay	Sets or queries whether serial bus signal analysis is to be performed.	5-178
:FLEXray?	Queries all FLEXRAY bus signal settings.	5-179
:FLEXray		
:ANALyze?	Queries all FLEXRAY bus signal settings.	5-179
[:ANALyze]		
:SETup?	Queries all FLEXRAY bus signal analysis settings.	5-179
:SETup		
:BCHannel	Sets or queries the channel bus type for FLEXRAY bus signal analysis.	5-179
:BRATe	Sets or queries the FLEXRAY bus signal analysis bit rate (data transfer rate).	5-179
:SOURce	Sets or queries the FLEXRAY bus signal analysis source.	5-179
:DETail?	Queries all FLEXRAY bus signal analysis result list settings.	5-179
:DETail		
:DISPlay	Sets or queries the display mode for the FLEXRAY bus signal analysis result list.	5-179
:LIST		
:ITEM?	Queries all items that will be displayed in the FLEXRAY bus signal analysis result list.	5-179
:VALue?	Queries all the data for the specified analysis number in the FLEXRAY bus signal analysis result list.	5-180
:SEARch?	Queries all FLEXRAY bus signal search settings.	5-180
:SEARch		
:ABORt	Aborts the FLEXRAY bus signal Search.	5-180
:EXECute	Executes a FLEXRAY bus signal Search.	5-180
:FJUMp		
:CCOunt	Jumps to the cycle count field in the FLEXRAY bus signal analysis result.	5-180
:CRC	Jumps to the CRC field in the FLEXRAY bus signal analysis result.	5-180
:FRAMeid	Jumps to the frame ID field in the FLEXRAY bus signal analysis result.	5-180
:HCRC	Jumps to the header CRC field in the FLEXRAY bus signal analysis result.	5-180
:PLENgth	Jumps to the payload length field in the FLEXRAY bus signal analysis result.	5-180
:SELect	Sets the detected waveform number for a FLEXRAY bus signal search and queries the zoom position corresponding to that number.	5-180
:SETup?	Queries all FLEXRAY bus signal search condition settings.	5-180
:SETup		
:ERRor?	Queries all FLEXRAY bus signal search error settings.	5-181
:ERRor		
:BSS	Sets or queries the FLEXRAY bus signal search BSS Error setting.	5-181
:CRC	Sets or queries the FLEXRAY bus signal search CRC Error setting.	5-181
:FES	Sets or queries the FLEXRAY bus signal search FES Error setting.	5-181
:HCRC	Sets or queries the FLEXRAY bus signal search Header CRC Error setting.	5-181
:IDData?	Queries all ID and data condition settings for FLEXRAY bus signal search.	5-181
[:IDData]		
:CCOunt?	Queries all FLEXRAY bus signal search cycle count settings.	5-181
:CCOunt		
:CONDition	Sets or queries the FLEXRAY bus signal search cycle count data condition.	5-181
:COUNt<y>	Sets or queries the FLEXRAY bus signal search cycle count setting.	5-182
:MODE	Sets or queries the cycle count mode, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-182
:DATA?	Queries the data, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-182
:DATA		
:BCOunt	Sets or queries the pattern comparison position, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-182

Command	Function	Page
:CONDition	Sets or queries the data comparison condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-182
:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:DECimal<y>	Sets or queries a reference value, which is one of the ID and data conditions, for FLEXRAY bus signal searching in decimal notation.	5-183
:DPATtern		
:HEXa	Sets a reference value, which is one of the ID and data conditions, for FLEXRAY bus signal searching in hexadecimal notation.	5-183
:PATTern	Sets or queries the data, which is one of the ID and data conditions, for FLEXRAY bus signal searching in binary notation.	5-183
:ENDian	Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:MODE	Sets or queries the data condition (enabled/disabled), which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:MSBLsb	Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:SIGN	Sets or queries the sign, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:FID?	Queries the frame ID, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:FID		
:CONDition	Sets or queries the frame ID data condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:ID<y>	Sets or queries the frame ID value, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:MODE	Sets or queries the frame ID condition (enabled/disabled), which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:INDIcator?	Queries the indicator, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-185
:INDIcator		
:MODE	Sets or queries the indicator condition (enabled/disabled), which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-185
:NFRame	Sets or queries the indicator null frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-185
:PPReamble	Sets or queries the indicator payload preamble, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-185
:STFRame	Sets or queries the indicator start frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-185
:SYFRame	Sets or queries the indicator sync frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-185
:MODE	Sets or queries the FLEXRAY bus signal search mode setting.	5-186
:I2C?	Queries all I2C bus signal analysis and search settings.	5-186
:I2C		
:ANALyze?	Queries all I2C bus signal analysis settings.	5-186
[:ANALyze]		
:SETup?	Queries all I2C bus signal analysis bus settings.	5-186
:SETup		
:CLOCk?	Queries all I2C bus signal analysis clock settings.	5-186
:CLOCk		
:SOURce	Sets or queries the I2C bus signal analysis clock.	5-186
:DATA?	Queries all I2C bus signal analysis data settings.	5-186
:DATA		
:SOURce	Sets or queries the I2C bus signal analysis data.	5-186
:INCLuderw	Sets or queries the enable/disable condition of address R/W for I2C bus signal analysis.	5-187
:DETail?	Queries all I2C bus signal analysis result list settings.	5-187
:DETail		
:DISPlay	Sets or queries the display mode for the I2C bus signal analysis result list.	5-187
:LIST		
:ITEM?	Queries all items that will be displayed in the I2C bus signal analysis result list.	5-187

Command	Function	Page
:VALue?	Queries all the data for the specified analysis number in the I2C bus signal analysis result list.	5-187
:SEARch?	Queries all I2C bus signal search settings.	5-187
:SEARch		
:ABORt	Aborts the I2C bus signal search.	5-187
:EXECute	Executes an I2C bus signal search.	5-187
:SELect	Sets which detected point to display in the I2C bus signal search zoom window and queries the zoom position of the detected point.	5-187
:SETup?	Queries all I2C bus signal search condition settings.	5-187
[:SETup]		
:ADDRess?	Queries all I2C bus signal search address pattern settings.	5-187
:ADDRess		
:ADDRess?	Queries all settings related to the address of the I2C bus signal search address pattern.	5-188
:ADDRess		
:BIT10ADdress?	Queries all I2C bus signal search 10-bit address settings.	5-188
:BIT10ADdress		
:DIRection	Sets or queries the 10-bit address data direction for I2C bus signal searching.	5-188
:HEXa	Sets the 10-bit address for I2C bus signal searching in hexadecimal notation.	5-188
:PATTern	Sets or queries the 10-bit address for I2C bus signal searching in binary notation.	5-188
[:BIT7ADdress]?	Queries all I2C bus signal search 7-bit address settings.	5-188
[:BIT7ADdress]		
:DIRection	Sets or queries the 7-bit address data direction for I2C bus signal searching.	5-188
:HEXa	Sets the 7-bit address for I2C bus signal searching in hexadecimal notation.	5-188
:PATTern	Sets or queries the 7-bit address for I2C bus signal searching in binary notation.	5-189
:BIT7APsub?	Queries all I2C bus signal search 7-bit + sub address settings.	5-189
:BIT7APsub		
:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit + sub address for I2C bus signal searching.	5-189
:ADDRess		
:DIRection	Sets or queries the 7-bit + sub address data direction for I2C bus signal searching.	5-189
:HEXa	Sets the 7-bit address of 7-bit + sub address for I2C bus signal searching in hexadecimal notation.	5-189
:PATTern	Sets or queries the 7-bit address of 7-bit + sub address for I2C bus signal searching in binary notation.	5-189
:SADDress?	Queries all settings related to the sub address of the 7-bit + sub address for I2C bus signal searching.	5-189
:SADDress		
:HEXa	Sets the sub address of 7-bit + sub address for I2C bus signal searching in hexadecimal notation.	5-189
:PATTern	Sets or queries the sub address of 7-bit + sub address for I2C bus signal searching in binary notation.	5-190
:MODE	Sets or queries the address condition enable/disable state for I2C bus signal searching.	5-190
:TYPE	Sets or queries the address condition address format for I2C bus signal searching.	5-190
:DATA?	Queries all settings related to the data of the I2C bus signal search address pattern.	5-190
:DATA		
:BCOunt	Sets or queries the data pattern comparison position for I2C bus signal searching.	5-190
:BMODe	Sets or queries on/off state of the data pattern comparison position for I2C bus signal searching.	5-190
:CONDition	Sets or queries the data judgment method (true/false) for I2C bus signal searching.	5-191
:DBYTe	Sets or queries the number of reference values for I2C bus signal searching.	5-191
:DPATtern		
:HEXa	Sets the data condition data for I2C bus signal searching in hexadecimal notation.	5-191

Command	Function	Page
:PATTern	Sets or queries the data condition data for I2C bus signal searching in binary notation.	5-191
:MODE	Sets or queries the data condition enable/disable state for I2C bus signal searching.	5-191
:GENeralcall?	Queries all I2C bus signal search general call settings.	5-191
:GENeralcall		
:BIT7Maddress?	Queries all settings related to the 7-bit master address of general call for I2C bus signal searching.	5-191
:BIT7Maddress		
:HEXa	Sets the 7-bit master address of general call for I2C bus signal searching in hexadecimal notation.	5-192
:PATTern	Sets or queries the 7-bit master address of general call for I2C bus signal searching in binary notation.	5-192
:SBYTe	Sets or queries the general call second byte type for I2C bus signal searching.	5-192
:MODE	Sets or queries the I2C bus signal search mode setting.	5-192
:NONack?	Queries all I2C bus signal search NON ACK ignore mode settings.	5-192
:NONack		
:HSMode	Sets or queries whether NONACK is to be ignored in high-speed mode of I2C bus signal searching.	5-192
:READaccess	Sets or queries whether NONACK is to be ignored in read access mode of I2C bus signal searching.	5-192
STARtbyte	Sets or queries whether NONACK is to be ignored in the start byte of I2C bus signal searching.	5-193
:LIN?	Queries all LIN bus signal analysis and search settings.	5-193
:LIN		
:ANALyze?	Queries all LIN bus signal analysis settings.	5-193
[:ANALyze]		
:SETup?	Queries all LIN bus signal analysis bus settings.	5-193
:SETup		
:BRATe	Sets or queries the LIN bus signal analysis bit rate (data transfer rate).	5-193
:REVision	Sets or queries the LIN bus signal analysis revision number.	5-193
:SOURce	Sets or queries the LIN bus signal analysis source.	5-193
:SPOint	Sets or queries the LIN bus signal analysis sample point.	5-193
:DETail?	Queries all LIN bus signal analysis result list settings.	5-193
:DETail		
:DISPlay	Sets or queries the display mode for the LIN bus signal analysis result list.	5-194
:LIST		
:ITEM?	Queries all items that will be displayed in the LIN bus signal analysis result list.	5-194
:VALue?	Queries all the data for the specified analysis number in the LIN bus signal analysis result list.	5-194
:SEARch?	Queries all LIN bus signal search settings.	5-194
:SEARch		
:ABORt	Aborts the LIN bus signal search.	5-194
:EXECute	Executes a LIN bus signal search.	5-194
:FJUMp		
:BREak	Jumps to the break field in the LIN bus signal search result.	5-194
:CSUM	Jumps to the checksum field in the LIN bus signal search result.	5-194
:DATA	Jumps to the data field in the LIN bus signal search result.	5-194
:IDENtifier	Jumps to the identifier field in the LIN bus signal search result.	5-194
:SYNCh	Jumps to the sync field in the LIN bus signal search result.	5-194
:SELect	Sets which detected point to display in the LIN bus signal search zoom window and queries the zoom position of the detected point.	5-194
:SELect?		
MAXimum	Queries the number of detected points in the LIN bus signal search.	5-195
:SETup?	Queries all LIN bus signal search condition settings.	5-195
:SETup		
:ERRor?	Queries all LIN bus signal search error settings.	5-195
:ERRor		
:CHECksum	Sets or queries the LIN bus signal search checksum error setting.	5-195
:FRAMing	Sets or queries the LIN bus signal search framing error setting.	5-195
:PARity	Sets or queries the LIN bus signal search parity error setting.	5-195

Command	Function	Page
:SYNCh	Sets or queries the LIN bus signal search synch error setting.	5-195
:TIMeout	Sets or queries the LIN bus signal search timeout error setting.	5-196
:IDData?	Queries all ID and data condition settings for LIN bus signal searching.	5-196
:IDData		
:DATA?	Queries all LIN bus signal search data settings.	5-196
:DATA		
:CONDition	Sets or queries the LIN bus signal search data condition setting.	5-196
:DBYTe	Sets or queries the number of data bytes for LIN bus signal searching.	5-196
:DECimal<y>	Sets or queries the data for LIN bus signal searching in decimal notation.	5-196
:DPATtern		
:HEXa	Sets the data for LIN bus signal searching in hexadecimal notation.	5-196
:PATTern	Sets or queries the data for LIN bus signal searching in binary notation.	5-197
:ENDian	Sets or queries the data endian for LIN bus signal searching.	5-197
:MODE	Sets or queries the data condition enable/disable state for LIN bus signal searching.	5-197
:MSBLsb	Sets or queries the MSB/LSB bit for LIN bus signal searching.	5-197
:SIGN	Sets or queries the data sign for LIN bus signal searching.	5-197
:IDENtifier?	Queries all LIN bus signal search identifier settings.	5-197
:IDENtifier		
:ID?	Queries all LIN bus signal search ID settings.	5-197
:ID		
:HEXa	Sets the ID for LIN bus signal searching in hexadecimal notation.	5-198
:MODE	Sets or queries the ID condition enable/disable state for LIN bus signal searching.	5-198
:PATTern	Sets or queries the ID for LIN bus signal searching in binary notation.	5-198
:MODE	Sets or queries the LIN bus signal search mode setting.	5-198
:RWINdow	Sets or queries the zoom window for showing the detected area.	5-198
:SENT?	Queries all SENT signal analysis and search settings.	5-198
:SENT		
:ANALyze?	Queries all SENT signal analysis settings.	5-198
[:ANALyze]		
:SETup?	Queries all bus setup settings for SENT signal analysis.	5-198
:SETup		
:DISPlay	Sets or queries the display mode for the SENT signal analysis result.	5-198
:FAST?	Queries all SENT signal analysis fast channel settings.	5-198
:FAST		
:DTYPe	Sets or queries the SENT signal analysis fast channel data format.	5-199
:USETup?	Queries all settings related to user-defined data of the SENT signal analysis fast channel.	5-199
:USETup		
:DATA<y>?	Queries all settings related to user-defined data of the SENT signal analysis fast channel.	5-199
:DATA<y>		
:MODE	Sets or queries the enable/disable condition of the specified user-defined data of the SENT signal analysis fast channel.	5-199
:ORDer	Sets or queries the byte order of the specified user-defined data of the SENT signal analysis fast channel.	5-199
:SIZE	Sets or queries the data size of the specified user-defined data of the SENT signal analysis fast channel.	5-199
:MULTiplexing	Sets or queries the multiplexing enable/disable condition of the specified user-defined data of the SENT signal analysis fast channel.	5-199
:FORMat?	Queries all SENT signal analysis format settings.	5-200
:FORMat		
:CEFactor?	Queries all SENT signal analysis error factor settings.	5-200
:CEFactor		
:SAComm?	Queries all status and communication error factor settings of SENT signal analysis.	5-200
:SAComm		
:BIT<y>?	Queries all status and communication error factor bit settings of SENT signal analysis.	5-200
:SCPulses	Sets or queries the consecutive calibration pulse error factor of SENT signal analysis.	5-200
:CRCType	Sets or queries the SENT signal analysis CRC computation type.	5-200

Command	Function	Page
:CTICk	Sets or queries the SENT signal analysis clock tick value.	5-200
:CTOLerance	Sets or queries the SENT signal analysis clock tolerance.	5-200
:DNIBbles	Sets or queries the number of data nibbles for SENT signal analysis.	5-201
:PPULse	Sets or queries the presence or absence of pause pulses for SENT signal analysis.	5-201
:VERSion	Sets or queries the SENT signal analysis specification version.	5-201
:STYPe	Sets or queries the SENT signal analysis slow channel format.	5-201
:SOURce	Sets or queries the SENT signal analysis source.	5-201
:DETail?	Queries all SENT signal analysis list settings.	5-201
:DETail		
:DISPlay	Sets or queries the display mode for the SENT signal analysis list.	5-201
:LIST		
:ALL?	Queries all the data for all analysis numbers in the SENT signal analysis result list.	5-202
:ITEM?	Queries all items that will be displayed in the SENT signal analysis result list.	5-202
:VALue?	Queries all the data for the specified analysis number in the SENT signal analysis result list.	5-202
:SEARch?	Queries all SENT signal search settings.	5-202
:SEARch		
:ABORt	Aborts the SENT signal search.	5-202
:EXECute	Executes a SENT signal search.	5-202
:SELect	Sets which detected point to display in the SENT signal search zoom window and queries the zoom position of the detected point.	5-202
:SELect?		
MAXimum	Queries the number of detected points in the SENT signal search.	5-202
:SETup?	Queries all SENT signal search setup settings.	5-202
[:SETup]		
:ERRor?	Queries all SENT signal search error settings.	5-202
:ERRor		
:FCRC	Sets or queries the SENT signal search fast channel CRC error.	5-203
:NDValue	Sets or queries the SENT signal search nibble data value error.	5-203
:NNUMber	Sets or queries the SENT signal search nibble data count error.	5-203
:SAComm	Sets or queries the SENT signal search status and communication error.	5-203
:SCPulses	Sets or queries the consecutive calibration pulse error for SENT signal searching.	5-203
:SCRC	Sets or queries the SENT signal search slow channel CRC error.	5-203
:FDATa?	Queries all fast channel data settings for SENT signal searching.	5-204
:FDATa		
:DATA<y>?	Queries all settings related to the specified fast channel user data for SENT signal searching.	5-204
:DATA<y>		
:CONDition	Sets or queries the comparison condition of the specified user data of the SENT signal search fast channel.	5-204
:DECimal<z>	Sets or queries the data of the specified user data of the SENT signal search fast channel in decimal notation.	5-204
:MODE	Sets or queries the data enable/disable condition of the specified user data of the SENT signal search fast channel.	5-204
:DNIBbles?	Queries all fast channel nibble data settings for SENT signal searching.	5-204
:DNIBbles		
:CONDition	Sets or queries the nibble data comparison condition of the SENT signal search fast channel.	5-204
:HEXa	Sets the SENT signal search fast channel nibble data in hexadecimal notation.	5-204
:PATTern	Sets or queries the SENT signal search fast channel nibble data in binary notation.	5-205
:FSAComm?	Queries all fast channel status and communication nibble settings for SENT signal searching.	5-205
:FSAComm		
:HEXa	Sets the SENT signal search fast channel status and communication nibble data in hexadecimal notation.	5-205
:PATTern	Sets or queries the SENT signal search fast channel status and communication nibble data in binary notation.	5-205
:MODE	Sets or queries the SENT signal search mode.	5-205

Command	Function	Page
:SDATa?	Queries all slow channel data settings for SENT signal searching.	5-205
:SDATa		
:ENHanced?	Queries all slow channel enhanced type settings for SENT signal searching.	5-205
:ENHanced		
:CBIT	Sets or queries the SENT signal search slow channel enhanced type configuration bit.	5-205
:D12Bit?	Queries all slow channel enhanced type 12-bit data and 8-bit ID settings for SENT signal searching.	5-205
:D12Bit		
:DATA?	Queries all slow channel enhanced type 12-bit data and 8-bit ID data settings for SENT signal searching.	5-206
:DATA		
:CONDition	Sets or queries the data comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.	5-206
:DECimal<y>	Sets or queries the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in decimal notation.	5-206
:HEXa	Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in hexadecimal notation.	5-206
:MODE	Sets or queries the data enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.	5-206
:PATTern	Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in binary notation.	5-206
:ID?	Queries all ID settings related to the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.	5-207
:ID		
:CONDition	Sets or queries the ID comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.	5-207
:DECimal<y>	Sets the ID of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching in decimal notation.	5-207
:MODE	Sets or queries the ID enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.	5-207
:D16Bit?	Queries all slow channel enhanced type 16-bit data and 4-bit ID settings for SENT signal searching.	5-207
:D16Bit		
:DATA?	Queries all slow channel enhanced type 16-bit data and 4-bit ID data settings for SENT signal searching.	5-207
:DATA		
:CONDition	Sets or queries the data comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.	5-207
:DECimal<y>	Sets or queries the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in decimal notation.	5-208
:HEXa	Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in hexadecimal notation.	5-208
:MODE	Sets or queries the data enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.	5-208
:PATTern	Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in binary notation.	5-208
:ID?	Queries all ID settings related to the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.	5-208
:ID		
:CONDition	Sets or queries the ID comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.	5-208
:DECimal<y>	Sets the ID of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching in decimal notation.	5-209
:MODE	Sets or queries the ID enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.	5-209
:SHORt?	Queries all slow channel short type settings for SENT signal searching.	5-209
:SHORt		
:DATA?	Queries all slow channel short type data settings for SENT signal searching.	5-209
:DATA		
:CONDition	Sets or queries the short type data comparison condition of the SENT signal search slow channel.	5-209
:DECimal<y>	Sets or queries the SENT signal search slow channel short type data in decimal notation.	5-209

Command	Function	Page
:HEXa	Sets the SENT signal search slow channel short type data in hexadecimal notation.	5-209
:MODE	Sets or queries the short type data enable/disable condition of the SENT signal search slow channel.	5-210
:PATTern	Sets the SENT signal search slow channel short type data in binary notation.	5-210
:ID?	Queries all slow channel short type ID settings for SENT signal searching.	5-210
:ID		
:CONDition	Sets or queries the short type ID comparison condition of the SENT signal search slow channel.	5-210
:DECimal<y>	Sets the SENT signal search slow channel short type ID in decimal notation.	5-210
:MODE	Sets or queries the short type ID enable/disable condition of the SENT signal search slow channel.	5-210
:TRENd<y>?	Queries all SENT signal analysis trend display settings.	5-210
:TRENd<y>		
:ASCale	Executes auto scaling of the SENT signal analysis trend display.	5-210
:CURSor?	Queries all SENT signal analysis trend display cursor measurement settings.	5-211
:CURSor		
:DT		
:VALue?	Queries the time value between cursors on the SENT signal analysis trend display.	5-211
:DV		
:VALue?	Queries the vertical value between cursors on the SENT signal analysis trend display.	5-211
:MODE	Sets or queries the automated measurement mode of the SENT signal analysis trend display.	5-211
:POSition<z>	Sets or queries the position of the specified cursor on the SENT signal analysis trend display.	5-211
:T<z>		
:VALue?	Queries the time value at the specified cursor on the SENT signal analysis trend display.	5-211
:V<z>		
:VALue?	Queries the vertical value at the specified cursor on the SENT signal analysis trend display.	5-211
:DISPlay	Sets or queries whether to show (ON) or hide (OFF) the SENT signal analysis trend.	5-211
:FCONtrol	Sets or queries the frame control value of the SENT signal analysis trend display.	5-212
:HRANge	Sets or queries the SENT signal analysis trend display source window.	5-212
:SID	Sets or queries the slow channel ID of the SENT signal analysis trend display in hexadecimal notation.	5-212
:SOURce	Sets or queries the SENT signal analysis trend display source channel.	5-212
:UDATa	Sets or queries the fast channel user-defined data of the SENT signal analysis trend display.	5-212
:VERTical	Sets or queries the vertical range of the SENT signal analysis trend display.	5-212
:VTDisplay	Sets or queries the on/off status of the VT waveform display on the SENT signal analysis trend display.	5-212
:SOURce?	Queries all analysis and search settings.	5-212
:SOURce		
:{CHANnel<y>|MATH<y>}?	Queries all source waveform settings.	5-213
:{CHANnel<y>|MATH<y>}		
:HYSTeresis	Sets or queries the source waveform hysteresis.	5-213
:LEVel	Sets or queries the source waveform level.	5-213
:SPATtern?	Queries all user-defined bus signal analysis and search settings.	5-213
:SPATtern		
:ANALyze?	Queries all user-defined bus signal analysis execution settings.	5-213
[:ANALyze]		
:SETup?	Queries all user-defined bus signal analysis settings.	5-213
:SETup		
:BRATe	Sets or queries the user-defined bus signal analysis bit rate.	5-213
:CLOCk?	Queries all user-defined bus signal analysis clock signal settings.	5-213
:CLOCk		
:MODE	Sets or queries the enable/disable condition of the user-defined bus signal analysis clock signal.	5-213

Command	Function	Page
:POLarity	Sets or queries the slope of the user-defined bus signal analysis clock signal.	5-214
:SOURce	Sets or queries the user-defined bus signal analysis clock signal.	5-214
:CS?	Queries all user-defined bus signal analysis chip select signal settings.	5-214
:CS		
:ACTive	Sets or queries the active state of the user-defined bus signal analysis chip select signal.	5-214
:SOURce	Sets or queries the user-defined bus signal analysis chip select signal.	5-214
:DATA?	Queries all user-defined bus signal analysis data signal settings.	5-214
:DATA		
:ACTive	Sets or queries the active state of the user-defined bus signal analysis data signal.	5-215
:SOURce	Sets or queries the user-defined bus signal analysis data signal.	5-215
:LATCh?	Queries all user-defined bus signal analysis latch signal settings.	5-215
:LATCh		
:POLarity	Sets or queries the slope of the user-defined bus signal analysis latch signal.	5-215
:SOURce	Sets or queries the user-defined bus signal analysis latch signal.	5-215
:SPOint	Sets or queries the analysis start point of the user-defined bus signal analysis.	5-215
:SEARch?	Queries all user-defined bus signal search settings.	5-216
:SEARch		
:ABORt	Aborts the user-defined bus signal search.	5-216
:EXECute	Executes a user-defined bus signal search.	5-216
:SELect	Sets which detected point to display in the user-defined bus signal search zoom window and queries the zoom position of the detected point.	5-216
:SELect?		
Maximum	Queries the number of detected points in the user-defined bus signal search.	5-216
:SETup?	Queries all User-defined bus signal search setup settings.	5-216
:SETup		
:BITSize	Sets or queries the user-defined bus signal search bit length.	5-216
:HEXa	Sets or queries the data conditions of the user-defined bus signal search in hexadecimal notation.	5-216
:PATTern	Sets or queries the data conditions of the user-defined bus signal search in binary notation.	5-216
:SPI?	Queries all SPI bus signal analysis and search settings.	5-216
:SPI		
:ANALyze?	Queries all SPI bus signal analysis settings.	5-216
[:ANALyze]		
:SETup?	Queries all SPI bus signal analysis bus settings.	5-217
:SETup		
:BITorder	Sets or queries the bit order of the SPI bus signal analysis data.	5-217
:CLOCk?	Queries all SPI bus signal analysis clock signal settings.	5-217
:CLOCk		
:POLarity	Sets or queries the slope of the SPI bus signal analysis clock signal.	5-217
:SOURce	Sets or queries the SPI bus signal analysis clock signal.	5-217
:CS?	Queries all SPI bus signal analysis chip select signal settings.	5-217
:CS		
:ACTive	Sets or queries the active state of the SPI bus signal analysis chip select signal.	5-217
:SOURce	Sets or queries the SPI bus signal analysis chip select signal.	5-218
:DATA<y>?	Queries all SPI bus signal analysis data signal settings.	5-218
:DATA<y>		
:SOURce	Sets or queries the SPI bus signal analysis data signal.	5-218
:FIELd	Sets or queries the field size of the SPI bus signal analysis data.	5-218
:GROuping	Sets or queries the grouping on/off state for when the SPI bus signal analysis chip select signal is not available.	5-218
:ITIMe	Sets or queries the idle time for when the SPI bus signal analysis chip select signal is not available.	5-218
:MODE	Sets or queries the wiring system (three-wire system or four-wire system) of the SPI bus signal analysis data signal.	5-219
:MSBLsb	Sets or queries the MSB/LSB bit for SPI bus signal analysis data.	5-219
:DETail?	Queries all SPI bus signal analysis result list settings.	5-219
:DETail		
:DISPlay	Sets or queries the display mode for the SPI bus signal analysis result list.	5-219

Command	Function	Page
:LIST		
:ITEM?	Queries all items that will be displayed in the SPI bus signal analysis result list.	5-219
:VALue?	Queries all the data for the specified analysis number in the SPI bus signal analysis result list.	5-219
:SEARch?	Queries all SPI bus signal search settings.	5-219
:SEARch		
:ABORt	Aborts the SPI bus signal search.	5-219
:EXECute	Executes an SPI bus signal search.	5-219
:SELect	Sets which detected point to display in the SPI bus signal search zoom window and queries the zoom position of the detected point.	5-220
:SELect?		
MAXimum	Queries the number of detected points in the SPI bus signal search.	5-220
:SETup?	Queries all SPI bus signal search setup settings.	5-220
[:SETup]		
:DATA<y>?	Queries all settings related to the specified data for SPI bus signal searching.	5-220
:DATA<y>		
:BCOunt	Sets or queries the pattern comparison start position of the specified data for SPI bus signal searching.	5-220
:CONDition	Sets or queries the judgment method (true/false) of the specified data for SPI bus signal searching.	5-220
:DBYTe	Sets or queries the data size (number of bytes) of the specified data for SPI bus signal searching.	5-220
:DPATtern		
:HEXa	Sets the specified data for SPI bus signal searching in hexadecimal notation.	5-221
:DPATtern		
:PATTern	Sets or queries the specified data for SPI bus signal searching in binary notation.	5-221
:MODE	Sets or queries the data condition enable/disable state for SPI bus signal searching.	5-221
:TYPE	Sets or queries the search type.	5-221
:UART?	Queries all UART signal analysis and search settings.	5-221
:UART		
:ANALyze?	Queries all UART signal analysis settings.	5-221
[:ANALyze]		
:SETup?	Queries all bus setup settings for UART signal analysis.	5-221
:SETup		
:BITorder	Sets or queries the UART signal analysis bit order.	5-221
:BRATe	Sets or queries the UART signal analysis bit rate (data transfer rate).	5-222
:BSPace	Sets or queries the UART signal analysis byte space.	5-222
:GROuping	Sets or queries on/off state of UART signal analysis grouping.	5-222
:POLarity	Sets or queries the UART signal analysis polarity.	5-222
:SOURce	Sets or queries the UART signal analysis signal.	5-222
:SPOint	Sets or queries the UART signal analysis sample point.	5-222
:DETail?	Queries all UART signal analysis result list settings.	5-223
:DETail		
:DISPlay	Sets or queries the display mode for the UART signal analysis result list.	5-223
:LIST		
:ITEM?	Queries all items that will be displayed in the UART signal analysis result list.	5-223
:VALue?	Queries all the data for the specified analysis number in the UART signal analysis result list.	5-223
:SEARch?	Queries all UART signal search settings.	5-223
:SEARch		
:ABORt	Aborts the UART signal search.	5-223
:EXECute	Executes a UART signal search.	5-223
:SELect	Sets which detected point to display in the UART signal search zoom window and queries the zoom position of the detected point.	5-223
:SELect?		
Maximum	Queries the number of detected points in the UART signal search.	5-223
:SETup?	Queries all search conditions for UART signal searching.	5-223
:SETup		
:DATA?	Queries all UART signal search data settings.	5-223
:DATA		

Command	Function	Page
:ASCii	Sets the UART signal search data in ASCII format.	5-224
:CONDition	Sets or queries the data judgment method (true/false) for UART signal searching.	5-224
:CSENsitive	Sets or queries whether to distinguish uppercase and lowercase letters in ASCII data for the UART signal search.	5-224
:DBYTe	Sets or queries the number of data bytes for UART signal searching.	5-224
:DPATtern		
:HEXa	Sets the data for UART signal searching in hexadecimal notation.	5-224
:DPATtern		
:PATTern	Sets or queries the data for UART signal searching in binary notation.	5-224
:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for UART signal searching.	5-225
:ERRor?	Queries all UART signal search error settings.	5-225
:ERRor		
:FRAMing	Sets or queries the UART signal search framing error setting.	5-225
:PARity	Sets or queries the UART signal search parity error setting.	5-225
:PMODe	Sets or queries the UART signal search parity mode setting.	5-225
:FORMat	Sets or queries the UART signal search format setting.	5-225
:MODE	Sets or queries the UART signal search mode setting.	5-225
:ZLINkage	Sets or queries the linkage (on/off) between the result numbers and zoom positions of the serial bus signal analysis results.	5-226

SNAP Group

:SNAP	Takes a snapshot.	5-227

SSTart Group

:SSTart?	Sets the trigger mode to single, starts waveform acquisition. The instrument returns 0 if the waveform acquisition is stopped within the specified timeout. Otherwise, the instrument returns 1.	5-228

STARt Group

:STARt	Starts waveform acquisition.	5-229

STATus Group

:STATus?	Queries all the settings of the communication status feature.	5-230
:STATus		
:CONDition?	Queries the contents of the condition register.	5-230
:EESE	Sets or queries the extended event enable register.	5-230
:EESR?	Queries the contents of the extended event register and clears the register.	5-230
:ERRor?	Queries the error code and message of the last error that has occurred (top of the error queue).	5-230
:FILTer<x>	Sets or queries transition filter.	5-230
:QENable	Sets or queries whether messages other than errors will be stored to the error queue.	5-230
:QMESsage	Sets or queries whether message information will be attached to the response to the STATus:ERRor? query.	5-230
:SPOLl?	Executes serial polling.	5-230

STOP Group

:STOP	Stops waveform acquisition.	5-231

STORe Group

:STORe?	Queries all the information related to setup data in the internal memory.	5-232
:STORe		
:SETup<x>?	Queries information about the setup data in the specified location of the internal memory.	5-232
:SETup<x>		
:COMMent	Sets or queries the comment for the setup data that is stored to the specified location in the internal memory.	5-232
:DATE?	Queries the date and time of the setup data that is stored to the specified location in the internal memory.	5-232

Command	Function	Page
:EXECute	Saves setup data to the specified location in the internal memory.	5-232
:LOCK	Sets or queries the protection on/off state for the setup data stored to the specified location in the internal memory.	5-232

SYSTem Group

:SYSTem?	Queries all system settings.	5-233
:SYSTem		
:BEEP	Generates a beep sound.	5-233
:CLICk	Sets or queries whether click sounds are produced.	5-233
:CLOCk?	Queries all date/time settings.	5-233
:CLOCk		
:DATE	Sets or queries the date.	5-233
:FORMat	Sets or queries the date format.	5-233
:MODE	Sets or queries whether the date and time are displayed.	5-233
:SNTP?	Queries the SNTP clock setting.	5-233
:SNTP		
:EXECute	Sets the clock using SNTP.	5-233
:GMTTime	Sets or queries the time difference from Greenwich Mean Time.	5-233
:TIME	Sets or queries the time.	5-233
:DCANcel	Sets or queries whether the specified delay is applied to time measurement values.	5-233
:LANGuage	Sets or queries the message language.	5-234
:LCD?	Queries all LCD settings.	5-234
:LCD		
:AUTO?	Queries all settings related to the LCD backlight auto-off feature.	5-234
:AUTO		
:MODE	Sets or queries on/off state of the LCD backlight auto-off feature.	5-234
:TIME	Sets or queries the LCD backlight auto-off time.	5-234
:BRIGhtness	Sets or queries the LCD brightness.	5-234
:MODE	Sets or queries the LCD backlight on/off state.	5-234
:MLANguage	Sets or queries the menu language.	5-234
:MPARameter?	Queries all settings related to automated measurement values of waveform parameters and cursor measurement values.	5-234
:MPARameter		
:FSIZe	Sets or queries the font size that is used to display the automatically measured waveform parameters and cursor measurement values.	5-234
:MROWs	Sets or queries the number of rows of automatically measured waveform parameters and cursor measurement values to show from the menu display area.	5-234
:OCANcel	Sets or queries whether the specified offset voltage is applied to measurement results and computation results.	5-234
:TOUT?	Queries all trigger out settings.	5-235
:TOUT		
:POLarity	Sets or queries the trigger out polarity.	5-235
:USBKeyboard	Sets or queries the USB keyboard type.	5-235

TIMebase Group

:TIMebase?	Queries all time base settings.	5-236
:TIMebase		
:SRATe?	Queries the sample rate.	5-236
:TDIV	Sets or queries the Time/div value.	5-236

TRIGger Group

:TRIGger?	Queries all trigger settings.	5-237
:TRIGger		
:ABN?	Queries all A->B(N) trigger settings.	5-237
:ABN		
:COUNt	Sets or queries the number of times condition B must be met for A->B(N) triggers.	5-237
:ACTion?	Queries all action-on-trigger settings.	5-237
:ACTion		

Command	Function	Page
:ACQCount	Sets or queries the action count of the action-on-trigger.	5-237
:BUZZer	Sets or queries whether a warning sound is generated as an action.	5-238
:HCOPy	Sets or queries whether screen capture data is output as an action.	5-238
:MAIL?	Queries all settings related to the mail notification on action.	5-238
:MAIL		
:COUNt	Sets or queries the upper limit to the number of mail notifications for the mail notification on action.	5-238
:MODE	Sets or queries whether a mail notification is made as an action.	5-238
:SAVE	Sets or queries whether waveform data is saved to the storage device as an action.	5-238
:STARt	Starts action-on-trigger.	5-238
:STOP	Stops action-on-trigger.	5-238
:ADB?	Queries all A Delay B trigger settings.	5-238
:ADB		
:DELay	Sets or queries the delay time for condition B for A Delay B triggers.	5-238
:{ATRigger|BTRigger}?	Queries all trigger condition settings.	5-238
{[:ATRigger]|:BTRigger}		
:CAN?	Queries all CAN bus signal trigger settings.	5-238
:CAN		
:BRATe	Sets or queries the CAN bus signal trigger bit rate (data transfer rate).	5-239
:EFRame?	Queries all CAN bus signal trigger error settings.	5-239
:EFRame		
:CRC	Sets or queries the CAN bus signal trigger CRC error setting.	5-239
[:MODE]	Sets or queries the CAN bus signal trigger error frame setting.	5-239
:STUFF	Sets or queries the CAN bus signal trigger stuff error setting.	5-239
:IDData?	Queries all ID and data condition settings for CAN bus signal triggering.	5-239
[:IDData]		
:ACK?	Queries the ACK setting, which is one of the ID and data conditions, for CAN bus signal triggering.	5-239
:ACK		
:MODE	Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN bus signal triggering.	5-239
:TYPE	Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-240
:DATA?	Queries the data, which is one of the ID and data conditions, for CAN bus signal triggering.	5-240
:DATA		
:CONDition	Sets or queries the data comparison condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-240
:DECimal<x>	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal triggering in decimal notation.	5-240
:DLC	Sets or queries the number of significant bytes (DLC), which is one of the ID and data conditions, for CAN bus signal triggering.	5-240
DPATtern		
:HEXa	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal triggering in hexadecimal notation.	5-240
:PATTern	Sets or queries the data, which is one of the ID and data conditions, for CAN bus signal triggering in binary notation.	5-240
:ENDian	Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN bus signal triggering.	5-241
:MODE	Sets or queries the data condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal triggering.	5-241
:MSBLsb	Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN bus signal triggering.	5-241
:SIGN	Sets or queries the data sign, which is one of the ID and data conditions, for CAN bus signal triggering.	5-241
:IDENtifier?	Queries the identifier, which is one of the ID and data conditions, for CAN bus signal triggering.	5-241
:IDENtifier		
:ID?	Queries the ID, which is one of the ID and data conditions, for CAN bus signal triggering.	5-241
:ID		

Command	Function	Page
:HEXa	Sets an ID, which is one of the ID and data conditions, for CAN bus signal triggering in hexadecimal notation.	5-241
:MODE	Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal triggering.	5-242
:PATTern	Sets or queries the ID condition, which is one of the ID and data conditions, for CAN bus signal triggering in binary notation.	5-242
:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN bus signal triggering.	5-242
:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CAN bus signal triggering.	5-242
:MSIGnal?	Queries the message signal, which is one of the ID and data conditions, for CAN bus signal triggering.	5-242
:MSIGnal		
:MESSage		
:ITEM	Sets the message item, which is one of the ID and data conditions, for CAN bus signal triggering.	5-242
:SELect	Sets or queries the CAN bus signal trigger message signal setting.	5-242
:SIGNal?	Queries the signal, which is one of the ID and data conditions, for CAN bus signal triggering.	5-243
:SIGNal		
:CONDition	Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-243
:DECimal<x>	Sets a reference signal value, which is one of the ID and data conditions, for CAN bus signal triggering in decimal notation.	5-243
:ITEM	Sets the signal item, which is one of the ID and data conditions, for CAN bus signal triggering.	5-243
:RTR	Sets or queries the RTR value, which is one of the ID and data conditions, for CAN bus signal triggering.	5-243
:IDOR?	Queries all ID OR condition settings for CAN bus signal triggering.	5-243
:IDOR		
:ACK?	Queries all ACK settings for the ID OR conditions for CAN bus signal triggering.	5-243
:ACK		
:MODE	Sets or queries the ACK mode, which is one of the ID OR conditions, for CAN bus signal triggering.	5-243
:TYPE	Sets or queries the ACK condition, which is one of the ID OR conditions, for CAN bus signal triggering.	5-243
:DATA?	Queries all data settings for the ID OR conditions for CAN bus signal triggering.	5-244
:DATA		
[:MODE]	Sets or queries the data enable/disable condition, which is one of the ID OR conditions, for CAN bus signal triggering.	5-244
:IDENtifier?	Queries all identifier settings for the ID OR conditions for CAN bus signal triggering.	5-244
:IDENtifier		
:ID<x>?	Queries all ID settings for the ID OR conditions for CAN bus signal triggering.	5-244
:ID<x>		
:HEXa	Sets the ID value, which is one of the ID OR conditions, for CAN bus signal triggering in hexadecimal notation.	5-244
:MODE	Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for CAN bus signal triggering.	5-244
:PATTern	Sets or queries the ID condition, which is one of the ID OR conditions, for CAN bus signal triggering in binary notation.	5-244
:MODE	Sets or queries the identifier enable/disable condition, which is one of the ID OR conditions, for CAN bus signal triggering.	5-244
:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the ID OR conditions, for CAN bus signal triggering.	5-245
:PFORmat	Sets or queries the ID input format, which is one of the ID OR conditions, for CAN bus signal triggering.	5-245
MSIGnal<x>		
:MESSage		

Command	Function	Page
:ITEM	Sets the message item, which is one of the ID OR conditions, for CAN bus signal triggering.	5-245
:RTR	Sets or queries the RTR, which is one of the ID OR conditions, for CAN bus signal triggering.	5-245
:MODE	Sets or queries the CAN bus signal trigger mode.	5-245
:RECessive	Sets or queries the CAN bus signal trigger recessive level.	5-245
:SOURce	Sets or queries the CAN bus signal trigger signal.	5-245
:SPOint	Sets or queries the CAN bus signal trigger sample point.	5-245
:CANFD?	Queries all CAN FD bus signal trigger settings.	5-246
:CANFD		
:BRATe	Sets or queries the CAN FD bus signal trigger bit rate (data transfer rate).	5-246
:DBRate	Sets or queries the CAN FD bus signal trigger data phase bit rate (data transfer rate).	5-246
:DSPoint	Sets or queries the CAN FD bus signal trigger data phase sample point.	5-246
:EFRame?	Queries all CAN FD bus signal trigger error settings.	5-246
:EFRame		
:CRC	Sets or queries the CAN FD bus signal trigger CRC error setting.	5-246
CRCEFactor?	Queries all CRC error factor settings for CAN FD bus signal triggering.	5-246
:CRCEFactor		
:CRCSequence	Sets or queries the CRC sequence (a CRC error factor) for CAN FD bus signal triggering.	5-246
:SCOunt	Sets or queries the Stuff Count (a CRC error factor) for CAN FD bus signal triggering.	5-247
:FSTuff	Sets or queries the CAN FD bus signal trigger fixed stuff error setting.	5-247
[:MODE]	Sets or queries the CAN FD bus signal trigger error frame setting.	5-247
:STUFF	Sets or queries the CAN FD bus signal trigger stuff error setting.	5-247
:FDF		
:CONDition	Sets or queries the CAN FD bus signal trigger FDF.	5-247
:FDSTandard	Sets or queries whether the CAN FD bus signal for triggering is an ISO standard signal.	5-247
:IDData?	Queries all ID and data condition settings for CAN FD bus signal triggering.	5-247
[:IDData]		
:ACK?	Queries all ACK settings for the ID and data conditions for CAN FD bus signal triggering.	5-248
:ACK		
:MODE	Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-248
:TYPE	Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-248
:DATA?	Queries all data settings for the ID and data conditions for CAN FD bus signal triggering.	5-248
:DATA		
:BCOunt	Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-248
:CONDition	Sets or queries the comparison condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-248
:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-248
:DECimal<x>	Sets a reference values (a, b), which is one of the ID and data conditions, for CAN FD bus signal triggering in decimal notation.	5-249
DPATtern		
:HEXa	Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal triggering in hexadecimal notation.	5-249
:PATTern	Sets or queries the data value, which is one of the ID and data conditions, for CAN FD bus signal triggering in binary notation.	5-249
:ENDian	Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-249
:MODE	Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-249
:MSBLsb	Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-249

Command	Function	Page
:SIGN	Sets or queries whether signed or unsigned data format will be used, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-250
IDENtifier?	Queries all identifier settings for the ID and data conditions for CAN FD bus signal triggering.	5-250
:IDENtifier		
:ID?	Queries all identifier settings for the ID and data conditions for CAN FD bus signal triggering.	5-250
:ID		
:HEXa	Sets the ID value, which is one of the ID and data conditions, for CAN FD bus signal triggering in hexadecimal notation.	5-250
:MODE	Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-250
:PATTern	Sets or queries the ID condition, which is one of the ID and data conditions, for CAN FD bus signal triggering in binary notation.	5-250
:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-250
:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-251
:MSIGnal?	Queries all message and signal settings for the ID and data conditions for CAN FD bus signal triggering.	5-251
:MSIGnal		
:MESSage		
:ITEM	Sets the message item, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-251
:SELect	Sets or queries the message and signal condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-251
:SIGNal?	Queries all signal settings for the ID and data conditions for CAN FD bus signal triggering.	5-251
:SIGNal		
:CONDition	Sets or queries the signal condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-251
:DECimal<x>	Sets a signal's reference value, which is one of the ID and data conditions, for CAN FD bus signal triggering in decimal notation.	5-251
:ITEM	Sets the signal item, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-251
:RTR	Sets or queries the RTR, which is one of the ID and data conditions, for CAN FD bus signal triggering.	5-252
:IDOR?	Queries all ID OR condition settings for CAN FD bus signal triggering.	5-252
:IDOR		
:ACK?	Queries all ACK settings for the ID OR conditions for CAN FD bus signal triggering.	5-252
:ACK		
:MODE	Sets or queries the ACK mode, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-252
:TYPE	Sets or queries the ACK condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-252
:DATA?	Queries all data settings for the ID OR conditions for CAN FD bus signal triggering.	5-252
:DATA		
[:MODE]	Sets or queries the data enable/disable condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-252
:IDENtifier?	Queries all identifier settings for the ID OR conditions for CAN FD bus signal triggering.	5-252
:IDENtifier		
:ID<x>?	Queries all ID settings for the ID OR conditions for CAN FD bus signal triggering.	5-252
:ID<x>		
:HEXa	Sets the ID value, which is one of the ID OR conditions, for CAN FD bus signal triggering in hexadecimal notation.	5-253
:MODE	Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-253
:PATTern	Sets or queries the ID condition, which is one of the ID OR conditions, for CAN FD bus signal triggering in binary notation.	5-253

Command	Function	Page
:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-253
:MODE	Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-253
:PFORmat	Sets or queries the ID input format, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-253
:MSIGnal<x>		
:MESSage		
:ITEM	Sets the message item, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-253
:RTR	Sets or queries the RTR, which is one of the ID OR conditions, for CAN FD bus signal triggering.	5-254
:MODE	Sets or queries the CAN FD bus signal trigger mode.	5-254
:RECessive	Sets or queries the CAN FD bus signal trigger recessive level.	5-254
:SOURce	Sets or queries the CAN FD bus signal trigger signal.	5-254
:SPOint	Sets or queries the CAN FD bus signal trigger sample point.	5-254
:CXPI?	Queries all CXPI bus signal trigger settings.	5-254
:CXPI		
:BRATe	Sets or queries the CXPI bus signal trigger bit rate (data transfer rate).	5-254
:CTOLerance	Sets or queries the CXPI bus signal trigger clock tolerance.	5-254
:ERROR?	Queries all CXPI bus signal trigger error settings.	5-255
:ERROR		
:CLOCk	Sets or queries the CXPI bus signal trigger clock error setting.	5-255
:CRC	Sets or queries the CXPI bus signal trigger CRC error setting.	5-255
:DLENgth	Sets or queries the CXPI bus signal trigger data length error setting.	5-255
:FRAMing	Sets or queries the CXPI bus signal trigger framing error setting.	5-255
:IBS	Sets or queries the CXPI bus signal trigger IBS error setting.	5-255
:PARity	Sets or queries the CXPI bus signal trigger parity error setting.	5-255
:IDData?	Queries all ID and data condition settings for CXPI bus signal triggering.	5-255
[:IDData]		
:DATA?	Queries the data, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-255
:DATA		
:BCOunt	Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-256
:CONDition	Sets or queries the data comparison condition, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-256
:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-256
:DECimal	Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal triggering in decimal notation.	5-256
:DPATtern		
:HEXa	Sets an ID, which is one of the ID and data conditions, for CXPI bus signal triggering in hexadecimal notation.	5-256
:PATTern	Sets or queries the ID condition, which is one of the ID and data conditions, for CXPI bus signal triggering in binary notation.	5-256
:ENDian	Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-257
:MODE	Sets or queries the ID and data condition (enabled/disabled) for CXPI bus signal triggering.	5-257
:MSBLsb	Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-257
:SIGN	Sets or queries the data sign, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-257
:FINFormation?	Queries all frame information settings for the ID and data conditions for CXPI bus signal triggering.	5-257
:FINFormation		
:CT	Sets or queries the frame information counter value, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-257
:MODE	Sets or queries the frame information (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal triggering.	5-257
:SLEEP	Sets or queries the frame information sleep bit, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-258

Command	Function	Page
:WAKeup	Sets or queries the frame information wakeup bit, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-258
:ID?	Queries the ID, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-258
:ID		
:HEXa	Sets an ID, which is one of the ID and data conditions, for CXPI bus signal triggering in hexadecimal notation.	5-258
:MODE	Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal triggering.	5-258
:PATTern	Sets or queries the ID, which is one of the ID and data conditions, for CXPI bus signal triggering in binary notation.	5-258
:PTYPE	Sets or queries the PTYPE comparison condition, which is one of the ID and data conditions, for CXPI bus signal triggering.	5-258
:IDOR?	Queries all CXPI bus signal trigger IDOR condition settings.	5-258
:IDOR		
:IDENtifier?	Queries all identifier settings for the IDOR conditions for CXPI bus signal triggering.	5-259
:IDENtifier		
:ID<x>?	Queries all ID settings for the IDOR conditions for CXPI bus signal triggering.	5-259
:ID<x>		
HEXa	Sets the ID value, which is one of the IDOR conditions, for CXPI bus signal triggering in hexadecimal notation.	5-259
MODE	Sets or queries the ID enable/disable condition, which is one of the IDOR conditions, for CXPI bus signal triggering.	5-259
PATTern	Sets or queries the ID condition, which is one of the IDOR conditions, for CXPI bus signal triggering in binary notation.	5-259
:MODE	Sets or queries the identifier enable/disable condition, which is one of the IDOR conditions, for CXPI bus signal triggering.	5-259
:PTYPE	Sets or queries the comparison condition of PTYPE, which is an IDOR condition, for CXPI bus signal triggering.	5-259
:MODE	Sets or queries the CXPI bus signal trigger type.	5-260
:PTYPE?	Queries all CXPI bus signal trigger PTYPE settings.	5-260
:PTYPE		
:CONDition	Sets or queries the CXPI bus signal trigger PTYPE condition setting.	5-260
:SOURce	Sets or queries the CXPI bus signal trigger signal.	5-260
:TSAMple	Sets or queries the logic value (1 or 0) determination threshold for CXPI bus signal triggering.	5-260
:WAKeupsleep?	Queries all wakeup and sleep settings for CXPI bus signal triggering.	5-260
:WAKeupsleep		
:TYPE	Sets or queries the wakeup and sleep conditions for CXPI bus signal triggering.	5-260
:FLEXray?	Queries all FlexRay bus signal trigger settings.	5-260
:FLEXray		
:BCHannel	Sets or queries the channel bus type for FlexRay bus signal triggering.	5-260
:BRATe	Sets or queries the FlexRay bus signal trigger bit rate (data transfer rate).	5-261
:ERRor?	Queries all FlexRay bus signal trigger error settings.	5-261
:ERRor		
:BSS	Sets or queries the FlexRay bus signal trigger BSS error setting.	5-261
:CRC	Sets or queries the FlexRay bus signal trigger CRC error setting.	5-261
:FES	Sets or queries the FlexRay bus signal trigger FES error setting.	5-261
IDData?	Queries all ID and data condition settings for FlexRay bus signal triggering.	5-261
[:IDData]		
:CCOunt?	Queries the cycle count, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-261
:CCOunt		
:CONDition	Sets or queries the cycle count data condition, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-261
:COUNt<x>	Sets or queries the cycle count, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-262
:MODE	Sets or queries the cycle count condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-262
:DATA<x>?	Queries the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-262

Command	Function	Page
:DATA<x>		
:BCOunt	Sets or queries the pattern comparison position of the data field data, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-262
:CONDition	Sets or queries the data field data condition, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-262
:DBYTe	Sets or queries the number of bytes of the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-262
:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for FlexRay bus signal triggering in hexadecimal notation.	5-263
DPATtern		
:HEXa	Sets a reference value for the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering in hexadecimal notation.	5-263
:PATTern	Sets or queries the data field data, which is one of the ID and data conditions, for FlexRay bus signal triggering in binary notation.	5-263
:ENDian	Sets or queries the data field endian, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-263
:MODE	Sets or queries the data field condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-263
:MSBLsb	Sets or queries the MSB/LSB bit of the data field data, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-263
:SIGN	Sets or queries the data sign of the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-264
:FID?	Queries the frame ID, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-264
:FID		
:CONDition	Sets or queries the frame ID data condition, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-264
:ID<x>	Sets or queries the frame ID value, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-264
:MODE	Sets or queries the frame ID condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-264
:INDIcator?	Queries the indicator, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-264
:INDIcator		
:MODE	Sets or queries the indicator condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-265
:NFRame	Sets or queries the indicator null frame, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-265
:PPReamble	Sets or queries the indicator payload preamble, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-265
:STFRame	Sets or queries the indicator start frame, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-265
:SYFRame	Sets or queries the indicator sync frame, which is one of the ID and data conditions, for FlexRay bus signal triggering.	5-265
:IDOR?	Queries all OR condition settings for FlexRay bus signal triggering.	5-265
:IDOR		
:ID<x>?	Queries all settings related to the specified frame ID of the OR condition for FlexRay bus signal triggering.	5-265
:ID<x>		
:CCOunt?	Queries all settings related to the cycle count of the specified frame ID of the OR condition for FlexRay bus signal triggering.	5-266
:CCOunt		
:CONDition	Sets or queries the cycle count of the specified frame ID of the OR condition for FlexRay bus signal triggering.	5-266
:COUNt<y>	Sets or queries the reference cycle count value of the specified frame ID of the OR condition for FlexRay bus signal triggering.	5-266
:FID?	Queries all settings related to the ID condition of the specified frame ID of the OR condition for FlexRay bus signal triggering.	5-266
:FID		
:CONDition	Sets or queries the ID judgment condition of the specified frame ID of the OR condition for FlexRay bus signal triggering.	5-266
:ID<y>	Sets or queries the ID reference value of the specified frame ID of the OR condition for FlexRay bus signal triggering.	5-266

Command	Function	Page
:MODE	Sets or queries the specified frame ID condition (enabled/disabled) of the OR condition for FlexRay bus signal triggering.	5-267
:MODE	Sets or queries the FlexRay bus signal trigger mode.	5-267
:SOURce	Sets or queries the FlexRay bus signal trigger source.	5-267
:I2C?	Queries all I2C bus signal trigger settings.	5-267
:I2C		
:ADDRess?	Queries all I2C bus signal trigger address pattern settings.	5-267
:ADDRess		
:ADDRess?	Queries all settings related to the address of the I2C bus signal trigger address pattern.	5-267
:ADDRess		
:BIT10ADdress?	Queries all I2C bus signal trigger 10-bit address settings.	5-267
:BIT10ADdress		
:DIRection	Sets or queries the 10-bit address data direction for I2C bus signal triggering.	5-267
:HEXa	Sets the 10-bit address for I2C bus signal triggering in hexadecimal notation.	5-267
:PATTern	Sets or queries the 10-bit address for I2C bus signal triggering in binary notation.	5-268
:BIT7ADdress?	Queries all I2C bus signal trigger 7-bit address settings.	5-268
[:BIT7ADdress]		
:DIRection	Sets or queries the 7-bit address data direction for I2C bus signal triggering.	5-268
:HEXa	Sets the 7-bit address for I2C bus signal triggering in hexadecimal notation.	5-268
:PATTern	Sets or queries the 7-bit address for I2C bus signal triggering in binary notation.	5-268
:BIT7APsub?	Queries all I2C bus signal trigger 7-bit + sub address settings.	5-268
:BIT7APsub		
:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit + sub address for I2C bus signal triggering.	5-268
:ADDRess		
:DIRection	Sets or queries the 7-bit + sub address data direction for I2C bus signal triggering.	5-268
:HEXa	Sets the 7-bit address of 7-bit + sub address for I2C bus signal triggering in hexadecimal notation.	5-269
:PATTern	Sets or queries the 7-bit address of 7-bit + sub address for I2C bus signal triggering in binary notation.	5-269
:SADDress?	Queries all settings related to the sub address of the 7-bit + sub address for I2C bus signal triggering.	5-269
:SADDress		
:HEXa	Sets the sub address of 7-bit + sub address for I2C bus signal triggering in hexadecimal notation.	5-269
:PATTern	Sets or queries the sub address of 7-bit + sub address for I2C bus signal triggering in binary notation.	5-269
:MODE	Sets or queries the address condition enable/disable state for I2C bus signal triggering.	5-269
:TYPE	Sets or queries the address condition address format for I2C bus signal triggering.	5-269
:DATA?	Queries all settings related to the data of the I2C bus signal trigger address pattern.	5-270
:DATA		
:BCOunt	Sets or queries the data pattern comparison position for I2C bus signal triggering.	5-270
:BMODe	Sets or queries on/off state of the data pattern comparison position for I2C bus signal triggering.	5-270
:CONDition	Sets or queries the data judgment method (true/false) for I2C bus signal triggering.	5-270
:DBYTe	Sets or queries the number of setup data values for I2C bus signal triggering.	5-270
DPATtern		
:HEXa	Sets the data for I2C bus signal triggering in hexadecimal notation.	5-270
:PATTern	Sets or queries the data condition data for I2C bus signal triggering in binary notation.	5-270
:MODE	Sets or queries the data condition enable/disable state for I2C bus signal triggering.	5-271
:GENeralcall?	Queries all I2C bus signal trigger general call settings.	5-271

Command	Function	Page
:GENeralcall		
:BIT7Maddress?	Queries all settings related to the 7-bit master address of general call for I2C bus signal triggering.	5-271
:BIT7Maddress		
:HEXa	Sets the 7-bit master address of general call for I2C bus signal triggering in hexadecimal notation.	5-271
:PATTern	Sets or queries the 7-bit master address of general call for I2C bus signal triggering in binary notation.	5-271
:SBYTe	Sets or queries the general call second byte type for I2C bus signal triggering.	5-271
:INCLuderw	Sets or queries the enable/disable condition of address R/W for I2C bus signal triggering.	5-271
:MODE	Sets or queries the I2C bus signal trigger mode.	5-271
:NONack?	Queries all I2C bus signal trigger NON ACK ignore mode settings.	5-271
:NONack		
:HSMode	Sets or queries whether NONACK is to be ignored in high-speed mode of I2C bus signal triggering.	5-272
:READaccess	Sets or queries whether NON ACK is to be ignored in the read access byte of I2C bus signal triggering.	5-272
:STARtbyte	Sets or queries whether NON ACK is to be ignored in the start byte of I2C bus signal triggering.	5-272
:SCL	Sets or queries the I2C bus signal trigger clock signal.	5-272
:SDA	Sets or queries the I2C bus signal trigger data signal.	5-272
:LIN?	Queries all LIN bus signal trigger settings.	5-272
:LIN		
:BLENgth	Sets or queries the LIN bus signal trigger break length.	5-272
:BRATe	Sets or queries the LIN bus signal trigger bit rate (data transfer rate).	5-273
:ERRor?	Queries all LIN bus signal trigger error settings.	5-273
:ERRor		
:PARity	Sets or queries the LIN bus signal trigger parity error setting.	5-273
:SYNCh	Sets or queries the LIN bus signal trigger synch error setting.	5-273
:IDData?	Queries all ID and data settings for LIN bus signal triggering.	5-273
:IDData		
:DATA?	Queries all LIN bus signal trigger data settings.	5-273
:DATA		
:CONDition	Sets or queries the LIN bus signal trigger data condition setting.	5-273
:DBYTe	Sets or queries the number of data bytes for LIN bus signal triggering.	5-273
:DECimal<x>	Sets the data for LIN bus signal triggering in decimal notation.	5-273
DPATtern		
:HEXa	Sets the data for LIN bus signal triggering in hexadecimal notation.	5-274
:PATTern	Sets or queries the data for LIN bus signal triggering in binary notation.	5-274
:ENDian	Sets or queries the data endian for LIN bus signal triggering.	5-274
:MODE	Sets or queries the data condition enable/disable state for LIN bus signal triggering.	5-274
:MSBLsb	Sets or queries the MSB/LSB bit for LIN bus signal triggering.	5-274
:SIGN	Sets or queries the data sign for LIN bus signal triggering.	5-274
:IDENtifier?	Queries all LIN bus signal trigger identifier settings.	5-274
:IDENtifier		
:ID?	Queries all LIN bus signal trigger ID settings.	5-275
:ID		
:HEXa	Sets the ID for LIN bus signal triggering in hexadecimal notation.	5-275
:MODE	Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for LIN bus signal triggering.	5-275
:PATTern	Sets or queries the ID for LIN bus signal triggering in binary notation.	5-275
:IDOR?	Queries all ID OR condition settings for LIN bus signal triggering.	5-275
:IDENtifier?	Queries all ID settings for the ID OR conditions for LIN bus signal triggering.	5-275
:IDENtifier		
:ID<x>?	Queries all settings related to the specified ID of the ID OR condition for LIN bus signal triggering.	5-275
:ID<x>		
:HEXa	Sets the ID value, which is one of the ID OR conditions, for LIN bus signal triggering in hexadecimal notation.	5-275

Command	Function	Page
:MODE	Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for LIN bus signal triggering.	5-275
:PATTern	Sets or queries the ID condition, which is one of the ID OR conditions, for LIN bus signal triggering in binary notation.	5-276
:MODE	Sets or queries the ID condition enable/disable state for LIN bus signal triggering.	5-276
:MODE	Sets or queries the LIN bus signal trigger mode.	5-276
:SOURce	Sets or queries the LIN bus signal trigger signal.	5-276
:SPOint	Sets or queries the LIN bus signal trigger sample point.	5-276
:OR?	Queries all edge OR trigger settings.	5-276
:OR		
:ALL	Sets or queries whether to make all channels edge OR trigger sources at once.	5-276
:CHANnel<x>	Sets or queries the slope of each channel of the edge OR trigger.	5-277
:PATTern?	Queries all settings related to the combination (pattern) trigger of multiple inputs.	5-277
:PATTern		
BITS	Queries all settings related to the logic inputs of the combination (pattern) trigger of multiple inputs.	5-277
BITS		
:HEXa	Sets the logic input state of the combination (pattern) trigger of multiple inputs in hexadecimal notation.	5-277
:PATTern	Sets or queries the logic input pattern of the combination (pattern) trigger of multiple inputs.	5-277
:CHANnel<x>	Sets or queries the pattern of the specified channel for the combination (pattern) trigger of multiple inputs.	5-277
:CLOCk?	Queries all settings related to the clock signal for the combination (pattern) trigger of multiple inputs.	5-277
:CLOCk		
:SLOPe	Sets or queries the clock signal slope for the combination (pattern) trigger of multiple inputs.	5-277
:SOURce	Sets or queries the clock signal for the combination (pattern) trigger of multiple inputs.	5-278
:CONDition	Sets or queries the trigger condition for the combination (pattern) trigger of multiple inputs.	5-278
:LOGic	Sets or queries the trigger combination condition for the combination (pattern) trigger of multiple inputs.	5-278
:TQUalify	Sets or queries the time condition for combination (pattern) triggering of multiple inputs.	5-278
:SENT?	Queries all SENT signal trigger settings.	5-278
:SENT		
:CEFactor?	Queries all SENT signal trigger error factor settings.	5-278
:CEFactor		
:SAComm?	Queries all status and communication error factor settings of SENT signal triggering.	5-278
:SAComm		
:BIT<x>	Sets or queries status and communication error factor bit of SENT signal triggering.	5-279
:SCPulses	Sets or queries the consecutive calibration pulse error factor of SENT signal triggering.	5-279
:CRCType	Sets or queries the SENT signal trigger CRC computation type.	5-279
:CTICk	Sets or queries the SENT signal trigger clock tick value.	5-279
:CTOLerance	Sets or queries the SENT signal trigger clock tolerance.	5-279
:DNIBbles	Sets or queries the number of data nibbles for SENT signal triggering.	5-279
:DTYPe	Sets or queries the SENT signal trigger fast channel data format.	5-279
:ERRor?	Queries all SENT signal trigger error settings.	5-280
:ERRor		
:FCRC	Sets or queries the SENT signal trigger fast channel CRC error.	5-280
:NDValue	Sets or queries the SENT signal trigger nibble data value error.	5-280
:NNUMber	Sets or queries the SENT signal trigger nibble data count error.	5-280
:SAComm	Sets or queries the SENT signal trigger status and communication error.	5-280
:SCRC	Sets or queries the SENT signal trigger slow channel CRC error.	5-280

Command	Function	Page
:SCPulses	Sets or queries the consecutive calibration pulse error for SENT signal triggering.	5-280
:FDATa?	Queries all fast channel data settings for SENT signal triggering.	5-280
:FDATa		
:DATA<x>?	Queries all settings related to the specified fast channel user data for SENT signal triggering.	5-281
:DATA<x>		
:CONDition	Sets or queries the comparison condition of the specified user data of the SENT signal trigger fast channel.	5-281
:DECimal<y>	Sets or queries the data of the specified user data of the SENT signal trigger fast channel in decimal notation.	5-281
:MODE	Sets or queries the data enable/disable condition of the specified user data of the SENT signal trigger fast channel.	5-281
:DNIBbles?	Queries all fast channel nibble data settings for SENT signal triggering.	5-281
:DNIBbles		
:CONDition	Sets or queries the nibble data comparison condition of the SENT signal trigger fast channel.	5-281
:HEXa	Sets the SENT signal trigger fast channel nibble data in hexadecimal notation.	5-281
:PATTern	Sets or queries the SENT signal trigger fast channel nibble data in binary notation.	5-282
:FSAComm?	Queries all fast channel status and communication nibble settings for SENT signal triggering.	5-282
:FSAComm		
:HEXa	Sets the SENT signal trigger fast channel status and communication nibble data in hexadecimal notation.	5-282
:PATTern	Sets or queries the SENT signal trigger fast channel status and communication nibble data in binary notation.	5-282
:MODE	Sets or queries the SENT signal trigger mode.	5-282
:PPULse	Sets or queries the presence or absence of pause pulses for SENT signal triggering.	5-282
:SDATa?	Queries all slow channel data settings for SENT signal triggering.	5-282
:SDATa		
:ENHanced?	Queries all slow channel enhanced type settings for SENT signal triggering.	5-282
:ENHanced		
:CBIT	Sets or queries the SENT signal trigger slow channel enhanced type configuration bit.	5-282
:D12Bit?	Queries all slow channel enhanced type 12-bit data and 8-bit ID settings for SENT signal triggering.	5-283
:D12Bit		
:DATA?	Queries all slow channel enhanced type 12-bit data and 8-bit ID data settings for SENT signal triggering.	5-283
:DATA		
:CONDition	Sets or queries the data comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.	5-283
:DECimal<x>	Sets or queries the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal triggering in decimal notation.	5-283
:HEXa	Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal triggering in hexadecimal notation.	5-283
:MODE	Sets or queries the data enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.	5-283
:PATTern	Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal triggering in binary notation.	5-283
:ID?	Queries all ID settings related to the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.	5-283
:ID		
:CONDition	Sets or queries the ID comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.	5-284
:DECimal<x>	Sets the ID of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering in decimal notation.	5-284
:MODE	Sets or queries the ID enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.	5-284

Command	Function	Page
:D16Bit?	Queries all slow channel enhanced type 16-bit data and 4-bit ID settings for SENT signal triggering.	5-284
:D16Bit		
:DATA?	Queries all slow channel enhanced type 16-bit data and 4-bit ID data settings for SENT signal triggering.	5-284
:DATA		
:CONDition	Sets or queries the data comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.	5-284
:DECimal<x>	Sets or queries the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal triggering in decimal notation.	5-284
:HEXa	Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal triggering in hexadecimal notation.	5-285
:MODE	Sets or queries the data enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.	5-285
:PATTern	Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal triggering in binary notation.	5-285
:ID?	Queries all ID settings related to the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.	5-285
:ID		
:CONDition	Sets or queries the ID comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.	5-285
:DECimal<x>	Sets the ID of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering in decimal notation.	5-285
:MODE	Sets or queries the ID enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.	5-285
:SHORt?	Queries all slow channel short type settings for SENT signal triggering.	5-286
:SHORt		
:DATA?	Queries all slow channel short type data settings for SENT signal triggering.	5-286
:DATA		
:CONDition	Sets or queries the short type data comparison condition of the SENT signal trigger slow channel.	5-286
:DECimal<x>	Sets or queries the SENT signal trigger slow channel short type data in decimal notation.	5-286
:HEXa	Sets the SENT signal trigger slow channel short type data in hexadecimal notation.	5-286
:MODE	Sets or queries the short type data enable/disable condition of the SENT signal trigger slow channel.	5-286
:PATTern	Sets the SENT signal trigger slow channel short type data in binary notation.	5-286
:ID?	Queries all slow channel short type ID settings for SENT signal triggering.	5-286
:ID		
:CONDition	Sets or queries the short type ID comparison condition of the SENT signal trigger slow channel.	5-287
:DECimal<x>	Sets the SENT signal trigger slow channel short type ID in decimal notation.	5-287
:MODE	Sets or queries the short type ID enable/disable condition of the SENT signal trigger slow channel.	5-287
:SOURce	Sets or queries the SENT signal trigger source.	5-287
:STYPe	Sets or queries the SENT signal trigger slow channel format.	5-287
:USETup?	Queries all settings related to user-defined data of the SENT signal trigger fast channel.	5-287
:USETup		
:DATA<x>?	Queries all settings related to user-defined data of the SENT signal trigger fast channel.	5-287
:DATA<x>		
:MODE	Sets or queries the enable/disable condition of the specified user-defined data of the SENT signal trigger fast channel.	5-288
:ORDer	Sets or queries the byte order of the specified user-defined data of the SENT signal trigger fast channel.	5-288
:SIZE	Sets or queries the data size of the specified user-defined data of the SENT signal trigger fast channel.	5-288
:MULTiplexing	Sets or queries the multiplexing enable/disable condition of the specified user-defined data of the SENT signal trigger fast channel.	5-288
:VERSion	Sets or queries the SENT signal trigger specification version.	5-288
:SIMPle?	Queries all edge trigger settings.	5-288

Command	Function	Page
:SIMPle		
:HFRejection	Sets or queries the on/off state of the low-pass filter (HF rejection) for the edge trigger source.	5-288
:LEVel	Sets or queries the trigger level of the edge trigger source.	5-289
:NREJection	Sets or queries the noise rejection of the edge trigger source level.	5-289
:PROBe	Sets or queries the probe of the external edge trigger source.	5-289
:RANGe	Sets or queries the input range of the external edge trigger source.	5-289
:SLOPe	Sets or queries the trigger slope (polarity if windows is ON) of the edge trigger source.	5-289
:SOURce	Sets or queries the edge trigger source.	5-289
:SPATtern?	Queries all user-defined bus signal trigger settings.	5-289
:SPATtern		
:BITSize	Sets or queries the user-defined bus signal trigger bit length.	5-290
:BRATe	Sets or queries the user-defined bus signal trigger bit rate.	5-290
:CLOCk?	Queries all user-defined bus signal trigger clock signal settings.	5-290
:CLOCk		
:MODE	Sets or queries the enable/disable condition of the user-defined bus signal trigger clock signal.	5-290
:POLarity	Sets or queries the polarity of the user-defined bus signal trigger clock signal.	5-290
:SOURce	Sets or queries the user-defined bus signal trigger clock signal.	5-290
:CS?	Queries all user-defined bus signal trigger chip select signal settings.	5-290
:CS		
:ACTive	Sets or queries the active state of the user-defined bus signal trigger chip select signal.	5-291
:SOURce	Sets or queries the user-defined bus signal trigger chip select signal.	5-291
:DATA?	Queries all user-defined bus signal trigger data signal settings.	5-291
:DATA		
:ACTive	Sets or queries the active state of the user-defined bus signal trigger data signal.	5-291
:SOURce	Sets or queries the user-defined bus signal trigger data signal.	5-291
:HEXa	Queries the data conditions of the user-defined bus signal trigger in hexadecimal notation.	5-291
:LATCh?	Queries all user-defined bus signal trigger latch signal settings.	5-291
:LATCh		
:POLarity	Sets or queries the polarity of the user-defined bus signal trigger latch signal.	5-292
:SOURce	Sets or queries the user-defined bus signal trigger latch signal.	5-292
:PATTern	Sets or queries the data conditions of the user-defined bus signal trigger in binary notation.	5-292
:SPI?	Queries all SPI bus signal trigger settings.	5-292
:SPI		
:BITorder	Sets or queries the bit order of the SPI bus signal trigger data.	5-292
:CLOCk?	Queries all SPI bus signal trigger clock signal settings.	5-292
:CLOCk		
:POLarity	Sets or queries the polarity of the SPI bus signal trigger clock signal.	5-292
:SOURce	Sets or queries the SPI bus signal trigger clock signal.	5-292
:CS?	Queries all SPI bus signal trigger chip select signal settings.	5-293
:CS		
:ACTive	Sets or queries the active state of the SPI bus signal trigger chip select signal.	5-293
:SOURce	Sets or queries the SPI bus signal trigger chip select signal.	5-293
:DATA<x>?	Queries all SPI bus signal trigger data signal settings.	5-293
:DATA<x>		
:BCOunt	Sets or queries the pattern comparison start position of the specified data for SPI bus signal triggering.	5-293
:CONDition	Sets or queries the judgment method (true/false) of the specified data for SPI bus signal triggering.	5-293
:DBYTe	Sets or queries the data size (number of bytes) of the specified data for SPI bus signal triggering.	5-293
DPATtern		
:HEXa	Sets the specified data for SPI bus signal triggering in hexadecimal notation.	5-294
:PATTern	Sets or queries the specified data for SPI bus signal triggering in binary notation.	5-294

Command	Function	Page
:MODE	Sets or queries the enable/disable condition of data signal for SPI bus signal triggering.	5-294
:SOURce	Sets or queries the SPI bus signal trigger data signal.	5-294
:MODE	Sets or queries the wiring system (three-wire system or four-wire system) of the SPI bus trigger.	5-294
:TYPE	Sets or queries the trigger type.	5-294
:UART?	Queries all UART signal trigger settings.	5-294
:UART		
:BITorder	Sets or queries the UART signal trigger bit order.	5-295
:BRATe	Sets or queries the UART signal trigger bit rate (data transfer rate).	5-295
:DATA?	Queries all UART signal trigger data settings.	5-295
:DATA		
:ASCii	Sets the UART signal trigger data in ASCII format.	5-295
:CONDition	Sets or queries the data judgment method (true/false) for UART signal triggering.	5-295
:CSENsitive	Sets or queries whether to distinguish uppercase and lowercase letters in ASCII data for UART signal triggering.	5-295
:DBYTe	Sets or queries the number of data bytes for UART signal triggering.	5-295
DPATtern		
:HEXa	Sets the data for UART signal triggering in hexadecimal notation.	5-296
:PATTern	Sets or queries the data for UART signal triggering in binary notation.	5-296
:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for UART signal triggering.	5-296
:ERRor?	Queries all UART signal trigger error settings.	5-296
:ERRor		
:FRAMing	Sets or queries the UART signal trigger framing error setting.	5-296
:PARity	Sets or queries the UART signal trigger parity error setting.	5-296
:PMODE	Sets or queries the UART signal trigger parity mode setting.	5-296
:FORMat	Sets or queries the UART signal trigger format setting.	5-297
:MODE	Sets or queries the trigger type for UART signal triggering.	5-297
:POLarity	Sets or queries the UART signal analysis polarity.	5-297
:SOURce	Sets or queries the UART signal analysis signal.	5-297
:SPOint	Sets or queries the UART signal analysis sample point.	5-297
:WINDow?	Queries all window trigger settings.	5-297
:WINDow		
:POLarity	Sets or queries the polarity of the window trigger source.	5-297
:SOURce	Sets or queries the window trigger source.	5-297
:TQUalify	Sets or queries the time condition for window triggering.	5-298
:WOR?	Queries all window OR trigger settings.	5-298
:WOR		
:ALL	Sets or queries whether to make all channels window OR trigger sources at once.	5-298
:CHANnel	Sets or queries the slope of each channel of the window OR trigger.	5-298
[:ATRigger]		
:INTerval?	Queries all edge interval trigger settings.	5-298
:INTerval		
:SLOPe	Sets or queries the edge interval trigger slope.	5-298
:SOURce	Sets or queries the edge interval trigger source.	5-298
:TQUalify	Sets or queries the time condition for edge interval triggering.	5-298
:PULSe?	Queries all pulse width trigger settings.	5-298
:PULSe		
:POLarity	Sets or queries the pulse width trigger source polarity.	5-299
:SOURce	Sets or queries the pulse width trigger source source.	5-299
:TQUalify	Sets or queries the time condition for pulse width triggering.	5-299
:RFTime?	Queries all rise/fall time trigger settings.	5-299
:RFTime		
:SLOPe	Sets or queries the rise/fall time trigger slope.	5-299
:SOURce	Sets or queries the rise/fall time trigger source.	5-299
:TQUalify	Sets or queries the time condition for rise/fall time triggering.	5-299
:RUNT?	Queries all runt trigger settings.	5-299
:RUNT		
:POLarity	Sets or queries the polarity of the runt trigger source.	5-299

Command	Function	Page
:SOURce	Sets or queries the runt trigger source.	5-299
:TQUalify	Sets or queries the time condition for runt triggering.	5-300
:TIMeout?	Queries all timeout period trigger settings.	5-300
:TIMeout		
:POLarity	Sets or queries the polarity of the timeout period trigger source.	5-300
:SOURce	Sets or queries the timeout period trigger source.	5-300
:TV?	Queries all TV trigger settings.	5-300
:TV		
:FIELd	Sets or queries the field that the TV trigger is to be activated on.	5-300
:FRAMe	Sets or queries the TV trigger frame skip function.	5-300
:{HDTV|NTSC|PAL|SDTV|USERdefine}?	Queries all TV trigger mode settings.	5-300
:{HDTV|NTSC|PAL|SDTV|USERdefine}		
:LINE	Sets or queries the line that the TV trigger is to be activated on.	5-300
:POLarity	Sets or queries the TV trigger input polarity.	5-301
:LEVel	Sets or queries the TV trigger level.	5-301
:LFORmat	Sets or queries the line designation method for TV triggering.	5-301
:SOURce	Sets or queries the TV trigger source.	5-301
:TYPE	Sets or queries the TV trigger input type.	5-301
:USERdefine?	Queries all user-defined TV trigger settings.	5-301
:USERdefine		
:DEFinition	Sets or queries the user-defined TV trigger definition.	5-301
:HFRejection	Sets or queries the on/off state of the low-pass filter (HF rejection) for user-defined TV triggering.	5-301
:HSYNc	Sets or queries the horizontal sync signal frequency for user-defined TV triggering.	5-302
:LINE	Sets or queries the line number for user-defined TV triggering.	5-302
:POLarity	Sets or queries the input polarity for user-defined TV triggering.	5-302
:SGUard	Sets or queries the sync guard for user-defined TV triggering.	5-302
:COMBination	Sets or queries the trigger combination.	5-302
:DELay?	Queries all delay settings.	5-302
:DELay		
:TIME	Sets or queries the delay (time from the trigger point to the trigger position).	5-302
:FORCe	Forces the instrument to trigger.	5-302
:HOLDoff?	Queries all hold-off settings.	5-303
:HOLDoff		
:TIME	Sets or queries the hold-off time.	5-303
:MODE	Sets or queries the trigger mode.	5-303
:POSition	Sets or queries trigger position.	5-303
:SCOunt	Sets or queries the number of times the trigger condition is to be met when the trigger mode is set to N Single.	5-303
:SOURce?	Queries all enhanced trigger source settings.	5-303
:SOURce		
:CHANnel<x>?	Queries all settings related to the specified trigger source for enhanced triggering.	5-303
:CHANnel<x>		
:HFRejection	Sets or queries the low-pass filter of the specified trigger source for enhanced triggering.	5-303
:NREJection	Sets or queries the noise rejection of the specified trigger source for enhanced triggering.	5-303
:LEVel	Sets or queries the trigger level of the specified trigger source for enhanced triggering.	5-303
ULLevel	Sets or queries the upper and lower limits of the enhanced trigger source level.	5-304
:TIME<y>	Sets or queries the time for enhanced triggering.	5-304

WAVeform Group

:WAVeform?	Queries all the waveform data information.	5-305
:WAVeform		
:ALL?	Queries all settings related to the :WAVeform:ALL:SEND? query.	5-305
:ALL		

Command	Function	Page
:SEND?	Queries the waveform data specified by :WAVeform:ALL:TRACe.	5-305
:TRACe	Sets or queries the target waveform for the :WAVeform:ALL:SEND? command.	5-306
:BITS?	Queries the bit length of the waveform data specified by :WAVeform:TRACe.	5-306
:BYTeorder	Sets or queries the transmission byte order for waveform data in word format, which is 2 bytes or longer in length.	5-306
:END	Sets or queries the end data point in the waveform specified by :WAVeform:TRACe.	5-306
:FORMat	Sets or queries the format of the waveform data to be sent.	5-306
:LENGth?	Queries the total number of points of the waveform specified by :WAVeform:TRACe.	5-306
:OFFSet?	Queries the offset value used to convert the waveform data specified by :WAVeform:TRACe to physical values.	5-306
:POSition?	Queries the vertical position used to convert waveform data into voltage when :WAVeform:FORMat is set to RBYTe.	5-306
:RANGe?	Queries the range value used to convert the waveform data specified by :WAVeform:TRACe to physical values.	5-306
:RECord	Sets or queries the record number that WAVeform commands will be applied to.	5-306
:RECord?		
MINimum	Queries the smallest record number in the source channel history.	5-307
:SEND?	Queries the waveform data specified by :WAVeform:TRACe.	5-307
:SIGN?	Queries whether signs are included in the binary data of the source waveform specified by :WAVeform:TRACe when the data is queried.	5-307
:SRATe?	Queries the sample rate of the record specified by :WAVeform:RECord.	5-307
:STARt	Sets or queries the start data point in the waveform specified by :WAVeform:TRACe.	5-307
:TRACe	Sets or queries the waveform that WAVeform commands will be applied to.	5-307
:TRIGger?	Queries the trigger position of the record specified by :WAVeform:RECord.	5-307
:TYPE?	Queries the acquisition mode of the waveform specified by :WAVeform:TRACe.	5-307

WPARameter Group

:WPARameter<x>?	Queries all waveform parameter measurement settings.	5-308
:WPARameter<x>		
:DISPlay	Sets or queries the on/off state of waveform parameter measurement display.	5-308
:HISTogram?	Queries all settings related to the histogram display of waveform parameter measurement.	5-308
:HISTogram		
:MEASure?	Queries all settings related to the automated measurement on the histogram display of waveform parameter measurement.	5-308
:MEASure		
:MODE	Sets or queries the auto measurement mode of the histogram display of waveform parameter measurement.	5-308
:PARameter?	Queries all settings related to the automated measurement of the histogram parameters of waveform parameter measurement.	5-308
:PARameter		
:ALL	Collectively turns on or off all the histogram parameters of waveform parameter measurement.	5-308
:<Parameter>?	Queries all settings related to the histogram parameters of waveform parameter measurement.	5-308
:<Parameter>		
:STATe	Sets or queries the on/off state the histogram parameters of waveform parameter measurement display.	5-308
:VALue?	Queries the measured value of a histogram parameter of waveform parameter measurement.	5-309
:POSition<y>	Sets or queries the position a parameter of waveform parameter measurement histogram.	5-309
:ITEM	Sets or queries a waveform parameter of waveform parameter measurement.	5-309
:MODE	Sets or queries the waveform parameter measurement mode.	5-309
:TRENd?	Queries all settings related to the trend display of waveform parameter measurement.	5-309
:TRENd		

Command	Function	Page
:ASCale	Executes auto scaling of the trend display of waveform parameter measurement.	5-310
:CURSor?	Queries all settings related to the cursor measurement on the trend display of waveform parameter measurement.	5-310
:CURSor		
:C<y>?	Queries the measured value of a cursor of waveform parameter measurement.	5-310
:DC?	Queries the measured value of a cursor of the trend of waveform parameter measurement.	5-310
:MODE	Sets or queries the auto measurement mode of the trend of waveform parameter measurement.	5-310
:POSition<y>	Sets or queries the position a cursor of the trend of waveform parameter measurement display.	5-310
:HRANge	Sets or queries the trend display source window for waveform parameter measurement.	5-310
:HSPan	Sets or queries the trend display horizontal span for waveform parameter measurement.	5-310
:VERTical	Sets or queries the vertical range of the trend of waveform parameter measurement.	5-310
:VTDisplay	Sets or queries the VT waveform display on/off state.	5-311
:WAIT?	Waits for the completion of automated measurement with a timeout.	5-311

XY Group

:XY<x>?	Queries all XY display settings.	5-312
:XY<x>		
:DISPlay	Sets or queries whether to show (ON) or hide (OFF) the XY display on the screen.	5-312
:MEASure?	Queries all automated measurement settings of XY display.	5-312
:MEASure		
:CURSor?	Queries all cursor measurement settings of XY display.	5-312
:CURSor		
:DX?	Queries all settings for the voltage difference between the XY display's horizontal cursors.	5-312
:DX		
:STATe	Sets or queries the on/off status of the voltage difference between the XY display's horizontal cursors.	5-312
:VALue?	Queries the voltage difference between the XY display's horizontal cursors.	5-312
:DY?	Queries all settings for the voltage difference between the XY display's vertical cursors.	5-312
:DY		
:STATe	Sets or queries the on/off status of the voltage difference between the XY display's vertical cursors.	5-312
:VALue?	Queries the voltage difference between the XY display's vertical cursors.	5-312
:X<y>?	Queries all horizontal cursor settings of XY display.	5-312
:X<y>		
:POSition	Sets or queries a horizontal cursor position on the XY display.	5-313
:STATe	Sets or queries the on/off status of the voltage of an XY display's horizontal cursor.	5-313
:VALue?	Queries the voltage at a XY display's horizontal cursor.	5-313
:Y<y>?	Queries all vertical cursor settings of XY display.	5-313
:Y<y>		
:POSition	Sets or queries a vertical cursor position on the XY display.	5-313
:STATe	Sets or queries the on/off status of the voltage of an XY display's vertical cursor.	5-313
:VALue?	Queries the voltage at a XY display's vertical cursor.	5-313
:INTeg?	Queries all integral settings of XY display.	5-313
:INTeg		
:LOOP	Sets or queries the integral method on the XY display.	5-313
:POLarity	Sets or queries the integral direction on the XY display.	5-314
:VALue?	Queries the integral value on the XY display.	5-314
:MODE	Sets or queries the automated measurement mode on the XY display.	5-314
RANGe	Sets or queries the VT waveform source window for the XY display.	5-314

Command	Function	Page
:SPLit	Sets or queries whether the XY1/XY2 display screen is split on the XY display.	5-314
:TRANge	Sets or queries the VT waveform range for the XY display.	5-314
:VTDisplay	Sets or queries the on/off state of the VT waveform display on the XY display.	5-314
:XTRace	Sets or queries the channel that is assigned to the X-axis on the XY display.	5-314
:YTRace	Sets or queries the channel that is assigned to the Y-axis on the XY display.	5-314

ZOOM Group

:ZOOM<x>?	Queries all waveform zoom settings.	5-315
:ZOOM<x>		
:ALLocation?	Queries all zoom source waveform settings.	5-315
:ALLocation		
:{CHANnel<y>|MATH<z>}	Sets or queries the zoom source waveform.	5-315
:DISPlay	Sets or queries the zoom waveform display on/off state.	5-315
:FORMat	Sets or queries the zoom waveform display format.	5-315
:MAG	Sets or queries the magnification of a zoom waveform display window.	5-315
:MAGFine	Sets or queries the magnification (fine) of a zoom waveform display window.	5-315
:MAIN	Sets or queries the zoom waveform display ratio.	5-315
:POSition	Sets or queries the position of a zoom box.	5-315
:VERTical?	Queries all vertical zoom settings.	5-315
:VERTical		
:{CHANnel<y>|MATH<z>}?	Queries all settings related to a vertical zoom trace.	5-316
:{CHANnel<y>|MATH<z>}		
:MAG	Sets or queries the vertical zoom factor.	5-316
:POSition	Sets or queries the vertical zoom position.	5-316
:TRACe	Sets or queries the trace to show on the vertical zoom window.	5-316

Common Command Group

*CAL	Executes calibration and queries the result.	5-317
*CLS	Clears the standard event register, extended event register, and error queue.	5-317
*ESE	Sets or queries the standard event enable register.	5-317
*ESR?	Queries and clears the standard event register.	5-317
*IDN?	Queries the instrument model.	5-317
*OPC	Sets bit 0 (the OPC bit) of the standard event register to 1 upon the completion of the specified overlap command.	5-317
*OPC?	When you send *OPC?, the instrument returns ASCII code 1 when the specified overlap command is completed.	5-318
*OPT?	Queries the installed options.	5-318
*RST	Initializes the settings.	5-318
*SRE	Sets or queries the service request enable register value.	5-318
*STB?	Queries the Status Byte Register value.	5-318
*TST?	Executes a self-test and queries the result. The self-test consists of tests of each kind of internal memory.	5-319
*WAI	Holds the execution of the subsequent command until the specified overlap command is completed.	5-319

5.2 ACQuire Group

:ACQuire?

Function Queries all waveform acquisition settings.

Syntax `:ACQuire?`

:ACQuire:AVERage?

Function Queries all settings related to averaging and waveform acquisition counts.

Syntax `:ACQuire:AVERage?`

:ACQuire:AVERage:COUNt

Function Sets or queries the attenuation constant of exponential averaging or the average count of simple averaging when in averaging mode.

Syntax `:ACQuire:AVERage:COUNt {<NRf>}`
<NRf> = 2 to 1024 (in 2n steps)

Example
```
:ACQUIRE:AVERAGE:COUNT 2
:ACQUIRE:AVERAGE:COUNT?
-> :ACQUIRE:AVERAGE:COUNT 2
```

:ACQuire:COUNt

Function Sets or queries the number of waveform acquisitions to perform in normal, envelope, and averaging modes.

Syntax `:ACQuire:COUNt {<NRf>|INFinity}`
<NRf> = 1 to 65536

Example
```
:ACQUIRE:COUNT 1
:ACQUIRE:COUNT?
-> :ACQUIRE:COUNT 1
```

:ACQuire:MODE

Function Sets or queries the waveform acquisition mode.

Syntax `:ACQuire:MODE {AVERage|ENVelope|NORMal}`

Example
```
:ACQUIRE:MODE AVERAGE
:ACQUIRE:MODE?
-> :ACQUIRE:MODE AVERAGE
```

:ACQuire:RESolution

Function Sets or queries the high resolution mode on/off state.

Syntax `:ACQuire:RESolution {<Boolean>}`

Example
```
:ACQUIRE:RESOLUTION ON
:ACQUIRE:RESOLUTION?
-> :ACQUIRE:RESOLUTION 1
```

Description When the high resolution mode is turned on, the maximum record length is halved.

:ACQuire:RLENgth

Function Sets or queries the record length.

Syntax `:ACQuire:RLENgth {<NRf>}`
<NRf> = See the Features Guide for this information.

Example
```
:ACQUIRE:RLENGTH 1250
:ACQUIRE:RLENGTH?
-> :ACQUIRE:RLENGTH 1250
```

:ACQuire:SAMPling

Function Sets or queries the sampling mode.

Syntax `:ACQuire:SAMPling {REAL|INTerporate|REPetitive}`

Example
```
:ACQUIRE:SAMPLING REAL
:ACQUIRE:SAMPLING?
-> :ACQUIRE:SAMPLING REAL
```

5.3 ANALysis Group

The power supply analysis feature (:ANALysis:PANalyze and :ANALysis:PMEAsure commands) is an option available only for 4-channel models. Power supply analysis commands cannot be used on 2-channel models or 4-channel models without the power supply analysis option.

:ANALysis?

Function Queries all analysis feature settings.

Syntax :ANALysis?

:ANALysis:AHIStogram<x>?

Function Queries all settings related to the histogram of the analysis feature.

Syntax :ANALysis:AHIStogram<x>?
<x> = 1 or 2 (1 only for 2-channel models)

:ANALysis:AHIStogram<x>:DISPlay

Function Sets or queries the waveform histogram display on/off state.

Syntax :ANALysis:AHIStogram<x>:
DISPlay {<Boolean>}
:ANALysis:AHIStogram<x>:DISPlay?
<x> = 1 or 2 (1 only for 2-channel models)

Example :ANALYSIS:AHISTOGRAM1:DISPLAY ON
:ANALYSIS:AHISTOGRAM1:DISPLAY?
-> :ANALYSIS:AHISTOGRAM1:DISPLAY 1

:ANALysis:AHIStogram<x>:HORizontal

Function Sets or queries the horizontal range of the waveform histogram.

Syntax :ANALysis:AHIStogram<x>:HORizont
al {<NRf>,<NRf>}
:ANALysis:AHIStogram<x>:HORizontal?
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = –4 div to 4 div

Example :ANALYSIS:AHISTOGRAM1:HORIZONTAL 0,1
:ANALYSIS:AHISTOGRAM1:HORIZONTAL?
-> :ANALYSIS:AHISTOGRAM1:HORIZONT
AL 1.000E+00,0.000E+00

:ANALysis:AHIStogram<x>:MEASure?

Function Queries all settings related to the waveform histogram measurement (including the mode on/off state).

Syntax :ANALysis:AHIStogram<x>:MEASure?
<x> = 1 or 2 (1 only for 2-channel models)

:ANALysis:AHIStogram<x>:MEASure:MODE

Function Sets or queries the waveform histogram measurement mode.

Syntax :ANALysis:AHIStogram<x>:MEASure:MO
DE {OFF|PARameter}
:ANALysis:AHIStogram<x>:MEASure:MO
DE?
<x> = 1 or 2 (1 only for 2-channel models)

Example :ANALYSIS:AHISTOGRAM1:MEASURE:MO
DE OFF
:ANALYSIS:AHISTOGRAM1:MEASURE:MODE?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:MO
DE OFF

:ANALysis:AHIStogram<x>:MEASure:PARameter?

Function Queries all settings related to the waveform histogram measurement when the mode is set to Param.

Syntax :ANALysis:AHIStogram<x>:MEASure:PARa
meter?
<x> = 1 or 2 (1 only for 2-channel models)

:ANALysis:AHIStogram<x>:MEASure:PARameter:ALL

Function Collectively turns on or off all the measurement items of the waveform histogram.

Syntax :ANALysis:AHIStogram<x>:MEASure:PARa
meter:ALL {<Boolean>}
<x> = 1 or 2 (1 only for 2-channel models)

Example :ANALYSIS:AHISTOGRAM1:MEASURE:PARAME
TER:ALL ON

:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>?

Function Queries all settings related to the specified measurement item of the waveform histogram.

Syntax :ANALysis:AHIStogram<x>:MEASure:PARa
meter:<Parameter>?
<x> = 1 or 2 (1 only for 2-channel models)
<Parameter> = {C1|C2|DC|MAXimum|MEAN|ME
Dian|MINimum|PEAK|SD2integ|SD3integ|SDEVi
ation|SDINteg}

:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>:STATe

Function Sets or queries the on/off state of the specified measurement item of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>:STATe {<Boolean>}
:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>:STATe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Parameter> = {C1|C2|DC|MAXimum|MEAN|MEDian|MINimum|PEAK|SD2integ|SD3integ|SDEViation|SDINteg}

Example The following example is for the maximum value.
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:MAXIMUM:STATE ON
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:MAXIMUM:STATE?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:MAXIMUM:STATE 1
```

:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>:VALue?

Function Queries the automatically measured value of the specified measurement item of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Parameter> = {C1|C2|DC|MAXimum|MEAN|MEDian|MINimum|PEAK|SD2integ|SD3integ|SDEViation|SDINteg}

Example The following example is for the maximum value.
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:MAXIMUM:VALUE?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:MAXIMUM:VALUE 1.000E+00
```

:ANALysis:AHIStogram<x>:MEASure:PARameter:HPOSition<y>

Function Sets or queries the horizontal position of Cursor1 or Cursor2 of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:MEASure:PARameter:HPOSition<y> {<NRf>}
:ANALysis:AHIStogram<x>:MEASure:PARameter:HPOSition<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = –5 div to 5 div

Example
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:HPOSITION1 1
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:HPOSITION1?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:HPOSITION 1.000E+00
```

:ANALysis:AHIStogram<x>:MEASure:PARameter:VPOSition<y>

Function Sets or queries the vertical position of Cursor1 or Cursor2 of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:MEASure:PARameter:VPOSition<y> {<NRf>}
:ANALysis:AHIStogram<x>:MEASure:PARameter:VPOSition<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = –4 div to 4 div

Example
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:VPOSITION1 1
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:VPOSITION1?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:VPOSITION 1.000E+00
```

:ANALysis:AHIStogram<x>:MODE

Function Sets or queries the target axis of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:MODE {HORizontal|VERTical}
:ANALysis:AHIStogram<x>:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:ANALYSIS:AHISTOGRAM1:MODE HORIZONTAL
:ANALYSIS:AHISTOGRAM1:MODE?
-> :ANALYSIS:AHISTOGRAM1:MODE HORIZONTAL
```

:ANALysis:AHIStogram<x>:RANGe

Function Sets or queries the measurement source window of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:RANGe {MAIN|Z1|Z2}
:ANALysis:AHIStogram<x>:RANGe?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:ANALYSIS:AHISTOGRAM1:RANGE MAIN
:ANALYSIS:AHISTOGRAM1:RANGE?
-> :ANALYSIS:AHISTOGRAM1:RANGE MAIN
```

:ANALysis:AHIStogram<x>:TRACe

Function Sets or queries the source waveform of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:TRACe {<NRf>|MATH<y>}
:ANALysis:AHIStogram<x>:TRACe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:ANALYSIS:AHISTOGRAM1:TRACE 1
:ANALYSIS:AHISTOGRAM1:TRACE?
-> :ANALYSIS:AHISTOGRAM1:TRACE 1
```

:ANALysis:AHIStogram<x>:VERTical

Function Sets or queries the vertical range of the waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:VERTic
al {<NRf>,<NRf>}
:ANALysis:AHIStogram<x>:VERTical?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = –4 div to 4 div

Example
```
:ANALYSIS:AHISTOGRAM1:VERTICAL 0,1
:ANALYSIS:AHISTOGRAM1:VERTICAL?
-> :ANALYSIS:AHISTOGRAM1:
VERTICAL 1.000E+00,0.000E+00
```

:ANALysis:PANalyze?

Function Queries all power supply analysis settings.

Syntax
```
:ANALysis:PANalyze?
```

:ANALysis:PANalyze:I2T?

Function Queries all Joule integral settings.

Syntax
```
:ANALysis:PANalyze:I2T?
```

:ANALysis:PANalyze:I2T:MATH

Function Sets or queries the Joule integral waveform display on/off state.

Syntax
```
:ANALysis:PANalyze:I2T:MATH {I2T|
OFF}
:ANALysis:PANalyze:I2T:MATH?
```

Example
```
:ANALYSIS:PANALYZE:I2T:MATH I2T
:ANALYSIS:PANALYZE:I2T:MATH?
-> :ANALYSIS:PANALYZE:I2T:MATH I2T
```

:ANALysis:PANalyze:I2T:MEASure?

Function Queries all settings related to the automated measurement of the Joule integral.

Syntax
```
:ANALysis:PANalyze:I2T:MEASure?
```

:ANALysis:PANalyze:I2T:MEASure:I2T?

Function Queries Joule integral settings.

Syntax
```
:ANALysis:PANalyze:I2T:MEASure:I2T?
```

:ANALysis:PANalyze:I2T:MEASure:I2T:COUNt?

Function Queries the continuous statistical processing count of the Joule integral.

Syntax
```
:ANALysis:PANalyze:I2T:MEASure:I2T:C
OUNt?
```

Example
```
:ANALYSIS:PANALYZE:I2T:MEASURE:I2T:C
OUNT?
-> :ANALYSIS:PANALYZE:I2T:MEASURE:I2
T:COUNT 100
```

:ANALysis:PANalyze:I2T:MEASure:I2T:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries the Joule integral statistics.

Syntax
```
:ANALysis:PANalyze:I2T:MEASure:I2T:{
MAXimum|MEAN|MINimum|SDEViation}?
```

Example
```
:ANALYSIS:PANALYZE:I2T:MEASURE:I2T:M
AXIMUM?
-> :ANALYSIS:PANALYZE:I2T:MEASURE:I2
T:MAXIMUM 10.0000E+03
```

Description If the statistical value cannot be obtained, the instrument returns “NAN” (not a number).

:ANALysis:PANalyze:I2T:MEASure:I2T:STATe

Function Sets or queries whether the Joule integral is to be measured.

Syntax
```
:ANALysis:PANalyze:I2T:MEASure:I2T:S
TATe {<Boolean>}
:ANALysis:PANalyze:I2T:MEASure:I2T:S
TATe?
```

Example
```
:ANALYSIS:PANALYZE:I2T:MEASURE:I2T:S
TATE ON
:ANALYSIS:PANALYZE:I2T:MEASURE:I2T:S
TATE?
-> :ANALYSIS:PANALYZE:I2T:MEASURE:I2
T:STATE 1
```

:ANALysis:PANalyze:I2T:MEASure:I2T:VALue?

Function Queries automatically measured value of the Joule integral.

Syntax
```
:ANALysis:PANalyze:I2T:MEASure:I2T:VALue? [{<NRf>}]
```
<NRf> = See the Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE:I2T:MEASURE:I2T:VALUE?
-> :ANALYSIS:PANALYZE:I2T:MEASURE:I2T:VALUE 10.0000E+03
```

Description
- If the value is immeasurable, the instrument returns “NAN” (not a number).
- The <NRf> is used to specify which iteration of automated measurement to query the measured value from.
- If <NRf> is set to 1, the oldest measured value in the automated measurement memory is queried.
- If a measured value does not exist at the specified iteration, the instrument returns “NAN” (not a number).
- If <NRf> is omitted, the most recent measured value is queried.
- If cyclic statistical processing for automated measurement is being executed and <NRf> is specified, the measured values over a cycle in iteration <NRf> from the left of the display on the displayed waveform is queried. If <NRf> is omitted, the measured values over the last cycle on the displayed waveform is queried.

:ANALysis:PANalyze:I2T:RANGe

Function Sets or queries the measurement source window.

Syntax
```
:ANALysis:PANalyze:I2T:RANGe {MAIN|Z1|Z2}
:ANALysis:PANalyze:I2T:RANGe?
```

Example
```
:ANALYSIS:PANALYZE:I2T:RANGE MAIN
:ANALYSIS:PANALYZE:I2T:RANGE?
-> :ANALYSIS:PANALYZE:I2T:RANGE MAIN
```

:ANALysis:PANalyze:I2T:SCALe?

Function Queries all scaling settings.

Syntax
```
:ANALysis:PANalyze:I2T:SCALe?
```

:ANALysis:PANalyze:I2T:SCALe:CENTer

Function Sets or queries the center value for manual scaling.

Syntax
```
:ANALysis:PANalyze:I2T:SCALe:CENTer {<NRf>}
:ANALysis:PANalyze:I2T:SCALe:CENTer?
```
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE:I2T:SCALE:CENTER 1
:ANALYSIS:PANALYZE:I2T:SCALE:CENTER?
-> :ANALYSIS:PANALYZE:I2T:SCALE:CENTER 1.00000E+00
```

:ANALysis:PANalyze:I2T:SCALe:MODE

Function Sets or queries the scale mode.

Syntax
```
:ANALysis:PANalyze:I2T:SCALe:MODE {AUTO|MANual}
:ANALysis:PANalyze:I2T:SCALe:MODE?
```

Example
```
:ANALYSIS:PANALYZE:I2T:SCALE:MODE AUTO
:ANALYSIS:PANALYZE:I2T:SCALE:MODE?
-> :ANALYSIS:PANALYZE:I2T:SCALE:MODE AUTO
```

:ANALysis:PANalyze:I2T:SCALe:SENSitivity

Function Sets or queries the sensitivity for manual scaling.

Syntax
```
:ANALysis:PANalyze:I2T:SCALe:SENSitivity {<NRf>}
:ANALysis:PANalyze:I2T:SCALe:SENSitivity?
```
<NRf> = 1.0000E–31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE:I2T:SCALE:SENSITIVITY 10
:ANALYSIS:PANALYZE:I2T:SCALE:SENSITIVITY?
-> :ANALYSIS:PANALYZE:I2T:SCALE:SENSITIVITY 10.0000E+00
```

:ANALysis:PANalyze:I2T:TRANge (TimeRange)

Function Sets or queries the measurement time period.

Syntax
```
:ANALysis:PANalyze:I2T:TRANge {<NRf>,<NRf>}
:ANALysis:PANalyze:I2T:TRANge?
```
<NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:ANALYSIS:PANALYZE:I2T:TRANGE -4,4
:ANALYSIS:PANALYZE:I2T:TRANGE?
-> :ANALYSIS:PANALYZE:I2T:TRANGE -4.00E+00,4.00E+00
```

:ANALysis:PANalyze:HARMonics?

Function Queries all harmonic analysis settings.

Syntax
```
:ANALysis:PANalyze:HARMonics?
```

:ANALysis:PANalyze:HARMonics:CCLass?

Function Queries all settings related to harmonic analysis class C.

Syntax
```
:ANALysis:PANalyze:HARMonics:CCLass?
```

:ANALysis:PANalyze:HARMonics:CCLass:GETLambda

Function Sets the current power factor of harmonic analysis class C.

Syntax
```
:ANALysis:PANalyze:HARMonics:CCLass:
GETLambda
```

Example
```
:ANALYSIS:PANALYZE:HARMONICS:CCLASS:
GETLAMBDA
```

:ANALysis:PANalyze:HARMonics:CCLass:LAMBda

Function Sets or queries the power factor of harmonic analysis class C.

Syntax
```
:ANALysis:PANalyze:HARMonics:CCLass:
LAMBda {<NRf>}
:ANALysis:PANalyze:HARMonics:CCLass:
LAMBda?
```
<NRf> = 0.01 to 1

Example
```
:ANALYSIS:PANALYZE:HARMONICS:CCLASS:
LAMBDA 0.10
:ANALYSIS:PANALYZE:HARMONICS:CCLASS:
LAMBDA?
-> :ANALYSIS:PANALYZE:HARMONICS:CCLA
SS:LAMBDA 100.0E-03
```

:ANALysis:PANalyze:HARMonics:CCLass:MAXCurrent

Function Sets or queries the fundamental current value of harmonic analysis class C.

Syntax
```
:ANALysis:PANalyze:HARMonics:CCLass:
MAXCurrent {<NRf>|<current>}
:ANALysis:PANalyze:HARMonics:CCLass:
MAXCurrent?
```
<NRf>, <current> = 0.001 to 100 (A)

Example
```
:ANALYSIS:PANALYZE:HARMONICS:CCLASS:
MAXCURRENT 50A
:ANALYSIS:PANALYZE:HARMONICS:CCLASS:
MAXCURRENT?
-> :ANALYSIS:PANALYZE:HARMONICS:CCLA
SS:MAXCURRENT 50.000E+00
```

:ANALysis:PANalyze:HARMonics:CCLass:OPOWer

Function Sets or queries whether active power 25 W of harmonic analysis class C will be exceeded.

Syntax
```
:ANALysis:PANalyze:HARMonics:CCLass:
OPOWer {FALSe|TRUE}
:ANALysis:PANalyze:HARMonics:CCLass:
OPOWer?
```

Example
```
:ANALYSIS:PANALYZE:HARMONICS:CCLASS:
OPOWER FALSE
:ANALYSIS:PANALYZE:HARMONICS:CCLASS:
OPOWER?
-> :ANALYSIS:PANALYZE:HARMONICS:CCLA
SS:OPOWER FALSE
```

:ANALysis:PANalyze:HARMonics:CLASs

Function Sets or queries the applicable class of the DUT in harmonic analysis.

Syntax
```
:ANALysis:PANalyze:HARMonics:CLA
Ss {A|B|C|D}
:ANALysis:PANalyze:HARMonics:CLASs?
```

Example
```
:ANALYSIS:PANALYZE:HARMONICS:CLASS A
:ANALYSIS:PANALYZE:HARMONICS:CLASS?
-> :ANALYSIS:PANALYZE:HARMONICS:CLA
SS A
```

:ANALysis:PANalyze:HARMonics:DCLass?

Function Queries all settings related to harmonic analysis class D.

Syntax
```
:ANALysis:PANalyze:HARMonics:DCLass?
```

:ANALysis:PANalyze:HARMonics:DCLass:POWer

Function Sets or queries the power value of harmonic analysis class D.

Syntax
```
:ANALysis:PANalyze:HARMonics:DCLass:
POWer {<NRf>}
:ANALysis:PANalyze:HARMonics:DCLass:
POWer?
```
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE:HARMONICS:DCLASS:
POWER 1V
:ANALYSIS:PANALYZE:HARMONICS:DCLASS:
POWER?
-> :ANALYSIS:PANALYZE:HARMONICS:DCLA
SS:POWER 1.0000E+00
```

:ANALysis:PANalyze:HARMonics:DMODe

Function Sets or queries the display mode of harmonic analysis.

Syntax
```
:ANALysis:PANalyze:HARMonics:DMO
De {LINear|LOG}
:ANALysis:PANalyze:HARMonics:DMODe?
```

Example
```
:ANALYSIS:PANALYZE:HARMONICS:DMO
DE LINEAR
:ANALYSIS:PANALYZE:HARMONICS:DMODE?
-> :ANALYSIS:PANALYZE:HARMONICS:DMO
DE LINEAR
```

:ANALysis:PANalyze:HARMonics:DETail?

Function Queries all settings related to the harmonic analysis result list.

Syntax
```
:ANALysis:PANalyze:HARMonics:DETail?
```

:ANALysis:PANalyze:HARMonics:DETail:DISPlay

Function Sets or queries the display mode of the harmonic analysis result list.

Syntax
```
:ANALysis:PANalyze:HARMonics:DETail:
DISPlay {FULL|LOWer|UPPer}
:ANALysis:PANalyze:HARMonics:DETail:
DISPlay?
```

Example
```
:ANALYSIS:PANALYZE:HARMONICS:DETAIL:
DISPLAY FULL
:ANALYSIS:PANALYZE:HARMONICS:DETAIL:
DISPLAY?
-> :ANALYSIS:PANALYZE:HARMONICS:DETA
IL:DISPLAY FULL
```

:ANALysis:PANalyze:HARMonics:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the harmonic analysis result list.

Syntax
```
:ANALysis:PANalyze:HARMonics:DETail:
LIST:ITEM?
```

Example
```
:ANALYSIS:PANALYZE:HARMONICS:DETAIL:
LIST:ITEM?
-> :ANALYSIS:PANALYZE:HARMONICS:DETA
IL:LIST:ITEM "Order.,Measure(A),Limi
t(A),Measure(%),Limit(%),Info,"
```

:ANALysis:PANalyze:HARMonics:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the harmonic analysis result list.

Syntax
```
:ANALysis:PANalyze:HARMonics:DETail:
LIST:VALue? {<NRf>}
```
<NRf> = 1 to 40 (harmonic order)

Example
```
:ANALYSIS:PANALYZE:HARMONICS:DETAIL:
LIST:VALUE? 2
-> :ANALYSIS:PANALYZE:HA
RMONICS:DETAIL:LIST:VAL
UE " 2, 0.031, 0.020,3.149, 2.000,
NG,"
```

:ANALysis:PANalyze:HARMonics:GROuping

Function Sets or queries the harmonic analysis grouping.

Syntax
```
:ANALysis:PANalyze:HARMonics:GROupi
ng {OFF|TYPE1|TYPE2}
:ANALysis:PANalyze:HARMonics:GROupi
ng?
```

Example
```
:ANALYSIS:PANALYZE:HARMONICS:GROUPI
NG OFF
:ANALYSIS:PANALYZE:HARMONICS:GROUPI
NG?
-> :ANALYSIS:PANALYZE:HARMONICS:GROU
PING OFF
```

:ANALysis:PANalyze:HARMonics:RMS

Function Queries the rms value harmonic analysis.

Syntax
```
:ANALysis:PANalyze:HARMonics:RMS?
```

Example
```
:ANALysis:PANalyze:HARMonics:RMS?
```

:ANALysis:PANalyze:HARMonics:SPOint

Function Sets or queries the computation start point of harmonic analysis.

Syntax
```
:ANALysis:PANalyze:HARMonics:SPOi
nt {<NRf>}
:ANALysis:PANalyze:HARMonics:SPOint?
```
<NRf> = –5 div to 5 div

Example
```
:ANALYSIS:PANALYZE:HARMONICS:SPOI
NT 1
:ANALYSIS:PANALYZE:HARMONICS:SPOINT?
-> :ANALYSIS:PANALYZE:HARMONICS:SPOI
NT 1.000E+00
```

:ANALysis:PANalyze:HARMonics:SVOLtage

Function Sets or queries the supply voltage of harmonic analysis.

Syntax
```
:ANALysis:PANalyze:HARMonics:SVOLtage {<NRf>|<voltage>}
:ANALysis:PANalyze:HARMonics:SVOLtage?
```
<NRf>, <voltage> = 90 V to 440 V

Example
```
:ANALYSIS:PANALYZE:HARMONICS:SVOLTAGE 230
:ANALYSIS:PANALYZE:HARMONICS:SVOLTAGE?
-> :ANALYSIS:PANALYZE:HARMONICS:SVOLTAGE 230.00000E+00
```

:ANALysis:PANalyze:HARMonics:THD

Function Queries the harmonic distortion factor of harmonic computation.

Syntax
```
:ANALysis:PANalyze:HARMonics:THD?
```

Example
```
:ANALysis:PANalyze:HARMonics:THD?
```

:ANALysis:PANalyze:SETup?

Function Queries all power supply analysis input settings.

Syntax
```
:ANALysis:PANalyze:SETup?
```

:ANALysis:PANalyze:SETup:ADESkew

Function Executes auto deskewing for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:ADESkew
```

Example
```
:ANALYSIS:PANALYZE:SETUP:ADESKEW
```

:ANALysis:PANalyze:SETup:I?

Function Queries all current input channel settings for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:I?
```

:ANALysis:PANalyze:SETup:I:DESKew

Function Sets or queries auto deskewing of the current input channel for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:I:DESKew {<time>}
:ANALysis:PANalyze:SETup:I:DESKew?
```
<time> = –1000.00 ns to 1000.00 ns (in 10 ps steps)

Example
```
:ANALYSIS:PANALYZE:SETUP:I:DESKEW 1NS
:ANALYSIS:PANALYZE:SETUP:I:DESKEW?
-> :ANALYSIS:PANALYZE:SETUP:I:DESKEW 1.000E-09
```

:ANALysis:PANalyze:SETup:I:INPut

Function Sets or queries the current input channels for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:I:INPut {2|4}
:ANALysis:PANalyze:SETup:I:INPut?
```

Example
```
:ANALYSIS:PANALYZE:SETUP:I:INPUT 2
:ANALYSIS:PANALYZE:SETUP:I:INPUT?
-> :ANALYSIS:PANALYZE:SETUP:I:INPUT 2
```

:ANALysis:PANalyze:SETup:I:PROBe

Function Queries all current-to-voltage conversion ratio settings of the probe connected to the current input channel for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:I:PROBe {C0_001|C0_002|C0_005|C0_01|C0_02|C0_05|C0_1|C0_2|C0_5|C1|C2|C5|C10|C20|C50|C100|C200|C500|C1000|C2000}}
:ANALysis:PANalyze:SETup:I:PROBe?
```

Example
```
:ANALYSIS:PANALYZE:SETUP:I:PROBE C0_001
:ANALYSIS:PANALYZE:SETUP:I:PROBE?
-> :ANALYSIS:PANALYZE:SETUP:I:PROBE C0_001
```

:ANALysis:PANalyze:SETup:U?

Function Queries all voltage input channel settings for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:U?
```

:ANALysis:PANalyze:SETup:U:DESKew

Function Sets or queries auto deskewing of the voltage input channel for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:U:DESKew {<time>}
:ANALysis:PANalyze:SETup:U:DESKew?
```
<time> = –1000.00 ns to 1000.00 ns (in 10 ps steps)

Example
```
:ANALYSIS:PANALYZE:SETUP:U:DESKEW 1NS
:ANALYSIS:PANALYZE:SETUP:U:DESKEW?
-> :ANALYSIS:PANALYZE:SETUP:U:DESKEW 1.000E-09
```

:ANALysis:PANalyze:SETup:U:INPut

Function Sets or queries the voltage input channels for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:U:INP
ut {1|3}
:ANALysis:PANalyze:SETup:U:INPut?
```

Example
```
:ANALYSIS:PANALYZE:SETUP:U:INPUT 1
:ANALYSIS:PANALYZE:SETUP:U:INPUT?
-> :ANALYSIS:PANALYZE:SETUP:U:INP
UT 1
```

:ANALysis:PANalyze:SETup:U:PROBe

Function Sets or queries the probe attenuation of the voltage input channels for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:U:PRO
Be {<NRf>}
:ANALysis:PANalyze:SETup:U:PROBe?
```
<NRf> = See the Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE:SETUP:U:PROBE 1
:ANALYSIS:PANALYZE:SETUP:U:PROBE?
-> :ANALYSIS:PANALYZE:SETUP:U:PRO
BE 1.000
```

:ANALysis:PANalyze:SETup:RTRace

Function Sets or queries the deskewing source trace for power supply analysis.

Syntax
```
:ANALysis:PANalyze:SETup:RTRace {I|
U}
:ANALysis:PANalyze:SETup:RTRace?
```

Example
```
:ANALYSIS:PANALYZE:SETUP:RTRACE I
:ANALYSIS:PANALYZE:SETUP:RTRACE?
-> :ANALYSIS:PANALYZE:SETUP:RTRACE I
```

:ANALysis:PANalyze:SOA?

Function Queries all settings related to the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA?
```

:ANALysis:PANalyze:SOA:CURSor?

Function Queries all settings related to the cursor measurement on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:CURSor?
```

:ANALysis:PANalyze:SOA:CURSor:X<x>?

Function Queries all settings related to the horizontal cursors on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:CURSor:X<x>?
```
<x> = 1 or 2

:ANALysis:PANalyze:SOA:CURSor:X<x>:POS ition

Function Sets or queries the horizontal cursor positions on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:CURSor:X<x>:P
OSition {<NRf>}
:ANALysis:PANalyze:SOA:CURSor:X<x>:P
OSition?
```
<x> = 1 or 2
<NRf> = –4 div to 4 div

Example
```
:ANALYSIS:PANALYZE:SOA:CURSOR:X1:POS
ITION 1
:ANALYSIS:PANALYZE:SOA:CURSOR:X1:POS
ITION?
-> :ANALYSIS:PANALYZE:SOA:CURSOR:X:P
OSITION 1.000E+00
```

:ANALysis:PANalyze:SOA:CURSor:X<x>:VAL ue?

Function Queries the horizontal cursor voltages on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:CURSor:X<x>:V
ALue?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE:SOA:CURSOR:X1:VAL
UE?
-> :ANALYSIS:PANALYZE:SOA:CURSOR:X1:
VALUE 1.000E+00
```

:ANALysis:PANalyze:SOA:CURSor:Y<x>?

Function Queries all settings related to the vertical cursors on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:CURSor:Y<x>?
```
<x> = 1 or 2

:ANALysis:PANalyze:SOA:CURSor:Y<x>:POS ition

Function Sets or queries the vertical cursor positions on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:CURSor:Y<x>:P
OSition {<NRf>}
:ANALysis:PANalyze:SOA:CURSor:Y<x>:P
OSition?
```
<x> = 1 or 2
<NRf> = –4 div to 4 div

Example
```
:ANALYSIS:PANALYZE:SOA:CURSOR:Y1:POS
ITION 1
:ANALYSIS:PANALYZE:SOA:CURSOR:Y1:POS
ITION?
-> :ANALYSIS:PANALYZE:SOA:CURSOR:Y:P
OSITION 1.000E+00
```

:ANALysis:PANalyze:SOA:CURSor:Y<x>:VALue?

Function Queries the vertical cursor voltages on the XY display for safe operating area analysis.

Syntax `:ANALysis:PANalyze:SOA:CURSor:Y<x>:VALue?`
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE:SOA:CURSOR:Y1:VALUE?
-> :ANALYSIS:PANALYZE:SOA:CURSOR:Y1:VALUE 1.000E+00
```

:ANALysis:PANalyze:SOA:MODE

Function Sets or queries the automated measurement mode on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:MODE {CURSor|OFF}
:ANALysis:PANalyze:SOA:MODE?
```

Example
```
:ANALYSIS:PANALYZE:SOA:MODE CURSOR
:ANALYSIS:PANALYZE:SOA:MODE?
-> :ANALYSIS:PANALYZE:SOA:MODE CURSOR
```

:ANALysis:PANalyze:SOA:TRANge (Time Range)

Function Sets or queries the VT waveform range to show on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:TRANge {<NRf>,<NRf>}
:ANALysis:PANalyze:SOA:TRANge?
```
<NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:ANALYSIS:PANALYZE:SOA:TRANGE -4,4
:ANALYSIS:PANALYZE:SOA:TRANGE?
-> :ANALYSIS:PANALYZE:SOA:TRANGE -4.00,4.00
```

:ANALysis:PANalyze:SOA:VTDisplay

Function Sets or queries the on/off state of the VT waveform display on the XY display for safe operating area analysis.

Syntax
```
:ANALysis:PANalyze:SOA:VTDisplay {<Boolean>}
:ANALysis:PANalyze:SOA:VTDisplay?
```

Example
```
:ANALYSIS:PANALYZE:SOA:VTDISPLAY ON
:ANALYSIS:PANALYZE:SOA:VTDISPLAY?
-> :ANALYSIS:PANALYZE:SOA:VTDISPLAY 1
```

:ANALysis:PANalyze:SWLoss?

Function Queries all switching loss settings.

Syntax `:ANALysis:PANalyze:SWLoss?`

Description Use the following command to set the reference levels (e.g., distal, mesial, proximal) for determining the total loss.
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal?
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:MODE
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:PERCent
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:UNIT
:MEASure:{CHANnel<x>|MATH<x>}:METHod
```

:ANALysis:PANalyze:SWLoss:CCALc

Function Sets or queries the total loss equation.

Syntax
```
:ANALysis:PANalyze:SWLoss:CCALc{RDS|VCE|WAVeform}
:ANALysis:PANalyze:SWLoss:CCALc?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:CCALc RDS
:ANALYSIS:PANALYZE:SWLOSS:CCALc?
-> :ANALYSIS:PANALYZE:SWLOSS:CCALc RDS
```

:ANALysis:PANalyze:SWLoss:DPROximal?

Function Queries all distal, mesial, and proximal settings.

Syntax `:ANALysis:PANalyze:SWLoss:DPROximal?`

Example `:ANALYSIS:PANALYZE:SWLOSS:DPROXIMAL?`

:ANALysis:PANalyze:SWLoss:DPROximal:MODE

Function Sets or queries the distal, mesial, and proximal point mode setting.

Syntax
```
:ANALysis:PANalyze:SWLoss:DPROximal:
MODE {PERCent|UNIT}
:ANALysis:PANalyze:SWLoss:DPROximal:
MODE?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:DPROXIMAL:
MODE PERCENT
:ANALYSIS:PANALYZE:SWLOSS:DPROXIMAL:
MODE?
-> :ANALYSIS:PANALYZE:SWLOSS:DPROXIM
AL:MODE PERCENT
```

:ANALysis:PANalyze:SWLoss:DPROximal:PERCent

Function Sets or queries the distal, mesial, and proximal points as percentages.

Syntax
```
:ANALysis:PANalyze:SWLoss:DPROximal:
PERCent {<NRf>,<NRf>,<NRf>}
:ANALysis:PANalyze:SWLoss:DPROximal:
PERCent?
```
<NRf>, <NRf>, <NRf> = 0 to 100 (%, in steps of 1)

Example
```
:ANALYSIS:PANALYZE:SWLOSS:DPROXIMAL:
PERCENT 10,50,90
:ANALYSIS:PANALYZE:SWLOSS:DPROXIMAL:
PERCENT?
-> :ANALYSIS:PANALYZE:SWLOSS:DPROXIM
AL:PERCENT 10,50,90
```

:ANALysis:PANalyze:SWLoss:DPROximal:UNIT

Function Sets or queries the distal, mesial, and proximal points as voltages.

Syntax
```
:ANALysis:PANalyze:SWLoss:DPROximal:
UNIT {<NRf>,<NRf>,<NRf>}
:ANALysis:PANalyze:SWLoss:DPROximal:
UNIT?
```
<NRf>, <NRf>, <NRf> = See the Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE:SWLOSS:DPROXIMAL:
UNIT -1,0,1
:ANALYSIS:PANALYZE:SWLOSS:DPROXIMAL:
UNIT?
-> :ANALYSIS:PANALYZE:SWLOSS:DPROXIM
AL:UNIT -1.0000000E+00,0.0000000E+00
,1.0000000E+00
```

:ANALysis:PANalyze:SWLoss:ILEVel

Function Sets or queries the current level for determining the loss zero period of total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:ILEV
el {<NRf>|<current>}
:ANALysis:PANalyze:SWLoss:ILEVel?
```
<NRf>, <current> = See the Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE:SWLOSS:ILEVEL 1
:ANALYSIS:PANALYZE:SWLOSS:ILEVEL?
-> :ANALYSIS:PANALYZE:SWLOSS:ILEV
EL 1.0000000E+00
```

:ANALysis:PANalyze:SWLoss:INDicator

Function Sets or queries the total loss measurement location indicator.

Syntax
```
:ANALysis:PANalyze:SWLoss:INDicat
or {<Boolean>}
:ANALysis:PANalyze:SWLoss:INDicator?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:INDICAT
OR OFF
:ANALYSIS:PANALYZE:SWLOSS:INDICATOR?
-> :ANALYSIS:PANALYZE:SWLOSS:INDICAT
OR OFF
```

:ANALysis:PANalyze:SWLoss:MATH

Function Sets or queries the power waveform display on/off state.

Syntax
```
:ANALysis:PANalyze:SWLoss:
MATH {OFF|POWer}
:ANALysis:PANalyze:SWLoss:MATH?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MATH OFF
:ANALYSIS:PANALYZE:SWLOSS:MATH?
-> :ANALYSIS:PANALYZE:SWLOSS:
MATH OFF
```

:ANALysis:PANalyze:SWLoss:MEASure?

Function Queries all settings related to the automated measurement of power supply parameters.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure?
```

:ANALysis:PANalyze:SWLoss:MEASure:<Parameter>?

Function Queries the setting of a power supply analysis parameter.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:<P
arameter>?
```
<Parameter> = {CCOunt|PON|PTOFf|PTON|PTOTal|WHON|WHTOFf|WHTON|WHTOTal}

:ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:COUNt?

Function Queries the continuous statistical processing count of the power supply analysis parameter.

Syntax :ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:COUNt?
<Parameter> = {CCOunt|PON|PTOFf|PTON|PTOTal|WHON|WHTOFf|WHTON|WHTOTal}

Example :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:COUNT?
-> :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:COUNT 100

:ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries a statistical value of a power supply analysis parameter.

Syntax :ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?
<Parameter> = {CCOunt|PON|PTOFf|PTON|PTOTal|WHON|WHTOFf|WHTON|WHTOTal}

Example :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:MAXIMUM?
-> :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:MAXIMUM 1.000E+00

Description If the statistical value cannot be obtained, the instrument returns “NAN” (not a number).

:ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:STATe

Function Sets or queries the on/off state of the power supply analysis parameter.

Syntax :ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:STATe {<Boolean>}
:ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:STATe?
<Parameter> = {CCOunt|PON|PTOFf|PTON|PTOTal|WHON|WHTOFf|WHTON|WHTOTal}

Example :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:STATE ON
:ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:STATE?
-> :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:STATE 1

:ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:VALue?

Function Queries the automatically measured value of a power supply parameter.

Syntax :ANALysis:PANalyze:SWLoss:MEASure:<Parameter>:VALue? [{<NRf>}]
<NRf> = See the Features Guide for this information.
<Parameter> = {CCOunt|PON|PTOFf|PTON|PTOTal|WHON|WHTOFf|WHTON|WHTOTal}

Example :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:VALUE?
-> :ANALYSIS:PANALYZE:SWLOSS:MEASURE:P:VALUE 10.0000E+03

Description
- If the value is immeasurable, the instrument returns “NAN” (not a number).
- The <NRf> is used to specify which iteration of automated measurement to query the measured value from.
- If <NRf> is set to 1, the oldest measured value in the automated measurement memory is queried.
- If a measured value does not exist at the specified iteration, the instrument returns “NAN” (not a number).
- If <NRf> is omitted, the most recent measured value is queried.
- If cyclic statistical processing for automated measurement is being executed and <NRf> is specified, the measured values over a cycle in iteration <NRf> from the left of the display on the displayed waveform is queried. If <NRf> is omitted, the measured values over the last cycle on the displayed waveform is queried.

:ANALysis:PANalyze:SWLoss:MEASure:CONTinuous?

Function Queries all settings related to the normal statistical processing of automatically measured total loss.

:ANALysis:PANalyze:SWLoss:MEASure:CONTinuous:RESTart

Function Restarts the normal statistical processing of automatically measured total loss.

Syntax :ANALysis:PANalyze:SWLoss:MEASure:CONTinuous:RESTart

Example :ANALYSIS:PANALYZE:SWLOSS:MEASURE:CONTINUOUS:RESTART

:ANALysis:PANalyze:SWLoss:MEASure:CONTinuous:TLCHange

Function Sets or queries whether the normal statistical processing of automatically measured total loss is to be restarted.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:CONTinuous:TLCHange {RESTart|IGNore}
:ANALysis:PANalyze:SWLoss:MEASure:CONTinuous:TLCHange?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MEASURE:CONTINUOUS:TLCHANGE RESTART
:ANALYSIS:PANALYZE:SWLOSS:MEASURE:CONTINUOUS:TLCHANGE?
-> :ANALYSIS:PANALYZE:SWLOSS:MEASURE:CONTINUOUS:TLCHANGE RESTART
```

:ANALysis:PANalyze:SWLoss:MEASure:CYCLe:ABORt

Function Aborts the execution of the cyclic statistical processing of automatically measured total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:CYCLe:ABORt
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MEASURE:CYCLE:ABORT
```

:ANALysis:PANalyze:SWLoss:MEASure:CYCLe:EXECute

Function Executes the cyclic statistical processing of automatically measured total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:CYCLe:EXECute
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MEASURE:CYCLE:EXECUTE
```

:ANALysis:PANalyze:SWLoss:MEASure:HISTory:ABORt

Function Aborts the execution of the statistical processing of the history waveform of automatically measured total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:HISTory:ABORt
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MEASURE:HISTORY:ABORT
```

:ANALysis:PANalyze:SWLoss:MEASure:HISTory:EXECute

Function Executes the statistical processing of the history waveform of automatically measured total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:HISTory:EXECute
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MEASure:HISTory:EXECute
```

:ANALysis:PANalyze:SWLoss:MEASure:MODE

Function Sets or queries the statistical processing type of automatically measured total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:MODE {OFF|CONTinuous|CYCLe|HISTory}
:ANALysis:PANalyze:SWLoss:MEASure:MODE?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MEASure:MODE OFF
:ANALYSIS:PANALYZE:SWLOSS:MEASure:MODE?
-> :ANALYSIS:PANALYZE:SWLOSS:MEASure:MODE OFF
```

:ANALysis:PANalyze:SWLoss:MEASure:ZLINkage

Function Sets or queries the linkage (on/off) between the result numbers and zoom positions of the results of automatically measured total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:MEASure:ZLINkage {<Boolean>}
:ANALysis:PANalyze:SWLoss:MEASure:ZLINkage?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:MEASure:ZLINKAGE ON
:ANALYSIS:PANALYZE:SWLOSS:MEASure:ZLINKAGE?
-> :ANALYSIS:PANALYZE:SWLOSS:MEASure:ZLINKAGE ON
```

:ANALysis:PANalyze:SWLoss:METhod

Function Sets or queries the calculation method for high and low points.

Syntax
```
:ANALysis:PANalyze:SWLoss:METhod {AUTO|MAXimum|HISTogram}
:ANALysis:PANalyze:SWLoss:METhod?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:METHOD AUTO
:ANALYSIS:PANALYZE:SWLOSS:METHOD?
-> :ANALYSIS:PANALYZE:SWLOSS:METHOD AUTO
```

:ANALysis:PANalyze:SWLoss:RANGe

Function Sets or queries the measurement source window.

Syntax
```
:ANALysis:PANalyze:SWLoss:RANGe {MAIN|Z1|Z2}
:ANALysis:PANalyze:SWLoss:RANGe?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:RANGE MAIN
:ANALYSIS:PANALYZE:SWLOSS:RANGE?
-> :ANALYSIS:PANALYZE:SWLOSS:RANGE MAIN
```

:ANALysis:PANalyze:SWLoss:RDS

Function Sets or queries the total loss on-resistance.

Syntax
```
:ANALysis:PANalyze:SWLoss:RDS {<NRf>}
:ANALysis:PANalyze:SWLoss:RDS?
```
<NRf> = 0 to 100 (in 1 mΩ steps)

Example
```
:ANALYSIS:PANALYZE:SWLOSS:RDS 1
:ANALYSIS:PANALYZE:SWLOSS:RDS?
-> :ANALYSIS:PANALYZE:SWLOSS:
RDS 1.000E+00
```

:ANALysis:PANalyze:SWLoss:SCALe?

Function Queries all scaling settings.

Syntax
```
:ANALysis:PANalyze:SWLoss:SCALe?
```

:ANALysis:PANalyze:SWLoss:SCALe:CENTer

Function Sets or queries the center value for manual scaling.

Syntax
```
:ANALysis:PANalyze:SWLoss:SCALe:CENTer {<NRf>}
:ANALysis:PANalyze:SWLoss:SCALe:CENTer?
```
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE:SWLOSS:SCALE:CENTER 1
:ANALYSIS:PANALYZE:SWLOSS:SCALE:CENTER?
-> :ANALYSIS:PANALYZE:SWLOSS:SCALE:CENTER 1.00000E+00
```

:ANALysis:PANalyze:SWLoss:SCALe:MODE

Function Sets or queries the scale mode.

Syntax
```
:ANALysis:PANalyze:SWLoss:SCALe:MODE {AUTO|MANual}
:ANALysis:PANalyze:SWLoss:SCALe:MODE?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:SCALE:MODE AUTO
:ANALYSIS:PANALYZE:SWLOSS:SCALE:MODE?
-> :ANALYSIS:PANALYZE:SWLOSS:SCALE:MODE AUTO
```

:ANALysis:PANalyze:SWLoss:SCALe:SENSitivity

Function Sets or queries the sensitivity for manual scaling.

Syntax
```
:ANALysis:PANalyze:SWLoss:SCALe:SENSitivity {<NRf>}
:ANALysis:PANalyze:SWLoss:SCALe:SENSitivity?
```
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE:SWLOSS:SCALE:SENSITIVITY 10
:ANALYSIS:PANALYZE:SWLOSS:SCALE:SENSITIVITY?
-> :ANALYSIS:PANALYZE:SWLOSS:SCALE:SENSITIVITY 10.0000E+00
```

:ANALysis:PANalyze:SWLoss:TRANge (Time Range)

Function Sets or queries the measurement time period.

Syntax
```
:ANALysis:PANalyze:SWLoss:TRANge {<NRf>,<NRf>}
:ANALysis:PANalyze:SWLoss:TRANge?
```
<NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:ANALYSIS:PANALYZE:SWLOSS:TRANGE -4,4
:ANALYSIS:PANALYZE:SWLOSS:TRANGE?
-> :ANALYSIS:PANALYZE:SWLOSS:TRANGE -4.00E+00,4.00E+00
```

:ANALysis:PANalyze:SWLoss:ULEVel

Function Sets or queries the voltage level for determining the loss calculation period of total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:ULEVel {<NRf>|<voltage>}
:ANALysis:PANalyze:SWLoss:ULEVel?
```
<NRf>, <voltage> = See the Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE:SWLOSS:ULEVEL 1V
:ANALYSIS:PANALYZE:SWLOSS:ULEVEL?
-> :ANALYSIS:PANALYZE:SWLOSS:ULEVEL 1.0000000E+00
```

:ANALysis:PANalyze:SWLoss:UNIT

Function Sets or queries the watthour unit.

Syntax
```
:ANALysis:PANalyze:SWLoss:UNIT {J|WH}
:ANALysis:PANalyze:SWLoss:UNIT?
```

Example
```
:ANALYSIS:PANALYZE:SWLOSS:UNIT WH
:ANALYSIS:PANALYZE:SWLOSS:UNIT?
-> :ANALYSIS:PANALYZE:SWLOSS:UNIT WH
```

:ANALysis:PANalyze:SWLoss:VCE

Function Sets or queries the collector-emitter saturation voltage of total loss.

Syntax
```
:ANALysis:PANalyze:SWLoss:
VCE {<NRf>|<voltage>}
:ANALysis:PANalyze:SWLoss:VCE?
```
<NRf>, <voltage> = 0 to 50 V (in 100 mV steps)

Example
```
:ANALYSIS:PANALYZE:SWLOSS:VCE 1V
:ANALYSIS:PANALYZE:SWLOSS:VCE?
-> :ANALYSIS:PANALYZE:SWLOSS:
VCE 1.0000000E+00
```

:ANALysis:PANalyze:TYPE

Function Sets or queries the power supply analysis type.

Syntax
```
:ANALysis:PANalyze:TYPE {HARMonics|
I2T|OFF|SOA|SWLoss}
:ANALysis:PANalyze:TYPE?
```

Example
```
:ANALYSIS:PANALYZE:TYPE HARMONICS
:ANALYSIS:PANALYZE:TYPE?
-> :ANALYSIS:PANALYZE:TYPE HARMONICS
```

:ANALysis:PMEAsure<x>?

Function Queries all power measurement settings.

Syntax
```
:ANALysis:PMEAsure<x>?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1?
-> :ANALYSIS:PMEASURE1
```

:ANALysis:PMEAsure:CYCMode

Function Sets or queries power measurement cycle mode.

Syntax
```
:ANALysis:PMEAsure:CYCMode {OFF|
NCYCle}
:ANALysis:PMEAsure:CYCMode?
```

Example
```
:ANALYSIS:PMEASURE:CYCMODE OFF
:ANALYSIS:PMEASURE:CYCMODE?
-> :ANALYSIS:PMEASURE:CYCMODE OFF
```

:ANALysis:PMEAsure:CYCTrace

Function Sets or queries the source waveform of power measurement cycle mode.

Syntax
```
:ANALysis:PMEAsure:CYCTrace {<NRf>|
MATH<x>}
:ANALysis:PMEAsure:CYCTrace?
```
<NRf> = 1 to 4
<x> = 1 to 4

Example
```
:ANALYSIS:PMEASURE:CYCTRACE 1
:ANALYSIS:PMEASURE:CYCTRACE?
-> :ANALYSIS:PMEASURE:CYCTRACE 1
```

:ANALysis:PMEAsure<x>:IDPRoximal?

Function Queries all distal, mesial, and proximal settings.

Syntax
```
:ANALysis:PMEAsure<x>:IDPRoximal?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:IDPROXIMAL?
```

:ANALysis:PMEAsure<x>:IDPRoximal:MODE

Function Sets or queries the distal, mesial, and proximal point mode setting.

Syntax
```
:ANALysis:PMEAsure<x>:IDPRoximal:MO
DE {PERCent|UNIT}
:ANALysis:PMEAsure<x>:IDPRoximal:MO
DE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:IDPROXIMAL:MO
DE PERCENT
:ANALYSIS:PMEASURE1:IDPROXIMAL:MODE?
-> :ANALYSIS:PMEASURE1:IDPROXIMAL:MO
DE PERCENT
```

:ANALysis:PMEAsure<x>:IDPRoximal:PERCent

Function Sets or queries the distal, mesial, and proximal points as percentages.

Syntax
```
:ANALysis:PMEAsure<x>:IDPRoximal:PER
Cent {<NRf>,<NRf>,<NRf>}
:ANALysis:PMEAsure<x>:IDPRoximal:PER
Cent?
```
<x> = 1 or 2
<NRf>, <NRf>, <NRf> = 0 to 100 (%, in steps of 1)

Example
```
:ANALYSIS:PMEASURE1:IDPROXIMAL:PERCE
NT 10,50,90
:ANALYSIS:PMEASURE1:IDPROXIMAL:PERCE
NT?
-> :ANALYSIS:PMEASURE1:IDPROXIMAL:PE
RCENT 10,50,90
```

:ANALysis:PMEAsure<x>:IDPRoximal:UNIT

Function Sets or queries the distal, mesial, and proximal points as voltages.

Syntax
```
:ANALysis:PMEAsure<x>:IDPRoximal:UN
IT {<NRf>,<NRf>,<NRf>}
:ANALysis:PMEAsure<x>:IDPRoximal:UN
IT?
```
<x> = 1 or 2
<NRf>, <NRf>, <NRf> = See the Features Guide for this information.

Example
```
:ANALYSIS:PMEASURE1:IDPROXIMAL:UN
IT -1,0,1
:ANALYSIS:PMEASURE1:IDPROXIMAL:UNIT?
-> :ANALYSIS:PMEASURE1:IDPROXIMAL:UN
IT -1.0000000E+00,0.0000000E+00,1.00
00000E+00
```

:ANALysis:PMEAsure<x>:IMEThod

Function Sets or queries the high and low points.

Syntax
```
:ANALysis:PMEAsure<x>:IMEThod {AUTO|MAXimum|HISTogram}
:ANALysis:PMEAsure<x>:IMEThod?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:IMETHOD AUTO
:ANALYSIS:PMEASURE1:IMETHOD?
-> :ANALYSIS:PMEASURE1:IMETHOD AUTO
```

:ANALysis:PMEAsure<x>:INDicator

Function Sets or queries the measurement location indicator.

Syntax
```
:ANALysis:PMEAsure<x>:INDicator <Parameter>
:ANALysis:PMEAsure<x>:INDicator?
```
<x> = 1 or 2
<Parameter> = {AH|AHABs|AHN|AHP|IAC|IAVGfreq|IDC|IMN|INPeak|IPPeak|IPTopeak|IRMN|IRMS|LAMBda|P|Q|S|UAC|UAVGfreq|UDC|UMN|UNPeak|UPPeak|UPTopeak|URMN|URMS|WH|WHABs|WHN|WHP|Z}

Example
```
:ANALYSIS:PMEASURE1:INDICATOR AH
:ANALYSIS:PMEASURE1:INDICATOR?
-> :ANALYSIS:PMEASURE1:INDICATOR AH
```

:ANALysis:PMEAsure<x>:MEASure?

Function Queries all settings related to the automated measurement of power measurement parameters.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:MEASURE?
```

:ANALysis:PMEAsure<x>:MEASure:<Parameter>?

Function Queries the power measurement parameter settings.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:<Parameter>?
```
<x> = 1 or 2
<Parameter> = {AH|AHABs|AHN|AHP|IAC|IAVGfreq|IDC|IMN|INPeak|IPPeak|IPTopeak|IRMN|IRMS|LAMBda|P|Q|S|UAC|UAVGfreq|UDC|UMN|UNPeak|UPPeak|UPTopeak|URMN|URMS|WH|WHABs|WHN|WHP|Z}

Example
```
:ANALYSIS:PMEASURE1:MEASURE:AH?
```

:ANALysis:PMEAsure<x>:MEASure:<Parameter>:COUNt?

Function Queries the normal statistical processing count of the power measurement parameter.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:COUNt?
```
<x> = 1 or 2
<Parameter> = {AH|AHABs|AHN|AHP|IAC|IAVGfreq|IDC|IMN|INPeak|IPPeak|IPTopeak|IRMN|IRMS|LAMBda|P|Q|S|UAC|UAVGfreq|UDC|UMN|UNPeak|UPPeak|UPTopeak|URMN|URMS|WH|WHABs|WHN|WHP|Z}

Example
```
:ANALYSIS:PMEASURE1:MEASURE:AH:COUNT?
```

:ANALysis:PMEAsure<x>:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries the statistical value of a power measurement parameter.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?
```
<x> = 1 or 2
<Parameter> = {AH|AHABs|AHN|AHP|IAC|IAVGfreq|IDC|IMN|INPeak|IPPeak|IPTopeak|IRMN|IRMS|LAMBda|P|Q|S|UAC|UAVGfreq|UDC|UMN|UNPeak|UPPeak|UPTopeak|URMN|URMS|WH|WHABs|WHN|WHP|Z}

Example
```
:ANALYSIS:PMEASURE1:MEASURE:AH:MAXIMUM?
```

:ANALysis:PMEAsure<x>:MEASure:<Parameter>:STATe

Function Sets or queries the on/off state of a power measurement parameter.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:STATe {<Boolean>}
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:STATe?
```
<x> = 1 or 2
<Parameter> = {AH|AHABs|AHN|AHP|IAC|IAVGfreq|IDC|IMN|INPeak|IPPeak|IPTopeak|IRMN|IRMS|LAMBda|P|Q|S|UAC|UAVGfreq|UDC|UMN|UNPeak|UPPeak|UPTopeak|URMN|URMS|WH|WHABs|WHN|WHP|Z}

Example
```
:ANALYSIS:PMEASURE1:MEASURE:AH:STATE ON
:ANALYSIS:PMEASURE1:MEASURE:AH:STATE?
-> :ANALYSIS:PMEASURE1:MEASURE:AH:STATE 1
```

:ANALysis:PMEAsure<x>:MEASure:<Parameter>:VALue?

Function Queries the automatically measured power measurement parameter value.

Syntax `:ANALysis:PMEAsure<x>:MEASure:<Parameter>:VALue?`
<x> = 1 or 2
<Parameter> = {AH|AHABs|AHN|AHP|IAC|IAVGfreq|IDC|IMN|INPeak|IPPeak|IPTopeak|IRMN|IRMS|LAMBda|P|Q|S|UAC|UAVGfreq|UDC|UMN|UNPeak|UPPeak|UPTopeak|URMN|URMS|WH|WHABs|WHN|WHP|Z}

Example `:ANALYSIS:PMEASURE1:MEASURE:AH:VALUE?`

:ANALysis:PMEAsure<x>:MEASure:ALL

Function Collectively turns on or off the power measurement parameters.

Syntax `:ANALysis:PMEAsure<x>:MEASure:ALL {<Boolean>}`
<x> = 1 or 2

Example `:ANALYSIS:PMEASURE1:MEASURE:ALL ON`

:ANALysis:PMEAsure<x>:MEASure:CONTinuous

Function Queries all settings related to the normal statistical processing of automated power measurement.
<x> = 1 or 2

:ANALysis:PMEAsure<x>:MEASure:CONTinuous:RESTart

Function Restarts the normal statistical processing of automated power measurement.

Syntax `:ANALysis:PMEAsure<x>:MEASure:CONTinuous:RESTart`
<x> = 1 or 2

Example `:ANALYSIS:PMEASURE1:MEASURE:CONTINUOUS:RESTART`

:ANALysis:PMEAsure<x>:MEASure:CONTinuous:TLCHange

Function Sets or queries whether the normal statistical processing of automated power measurement is to be restarted.

Syntax `:ANALysis:PMEAsure<x>:MEASure:CONTinuous:TLCHange {RESTart|IGNore}`
`:ANALysis:PMEAsure<x>:MEASure:CONTinuous:TLCHange?`
<x> = 1 or 2

Example `:ANALYSIS:PMEASURE1:MEASURE:CONTINUOUS:TLCHANGE RESTART`
`:ANALYSIS:PMEASURE1:MEASURE:CONTINUOUS:TLCHANGE`
`-> :ANALYSIS:PMEASURE1:MEASURE:CONTINUOUS:TLCHANGE RESTART`

:ANALysis:PMEAsure<x>:MEASure:CYCLe

Function Queries all settings related to the cyclic statistical processing of automated power measurement.
<x> = 1 or 2

:ANALysis:PMEAsure<x>:MEASure:CYCLe:ABORt

Function Aborts the execution of the cyclic statistical processing of automated power measurement.

Syntax `:ANALysis:PMEAsure<x>:MEASure:CYCLe:ABORt`
<x> = 1 or 2

Example `:ANALYSIS:PMEASURE1:MEASURE:CYCLE:ABORT`

:ANALysis:PMEAsure<x>:MEASure:CYCLe:EXECute

Function Executes the cyclic statistical processing of automated power measurement.

Syntax `:ANALysis:PMEAsure<x>:MEASure:CYCLe:EXECute`
<x> = 1 or 2

Example `:ANALYSIS:PMEASURE1:MEASURE:CYCLE:EXECUTE`

:ANALysis:PMEAsure<x>:MEASure:CYCLe:TRACe

Function Sets or queries the source waveform of the cyclic statistical processing of automated power measurement.

Syntax `:ANALysis:PMEAsure<x>:MEASure:CYCLe:TRACe {<NRf>|MATH<y>}`
`:ANALysis:PMEAsure<x>:MEASure:CYCLe:TRACe?`
<x> = 1 or 2
<NRf> = 1 to 4
<y> = 1 to 4

Example `:ANALYSIS:PMEASURE1:MEASURE:CYCLE:TRACE 1`
`:ANALYSIS:PMEASURE1:MEASURE:CYCLE:TRACE?`
`-> :ANALYSIS:PMEASURE1:MEASURE:CYCLE:TRACE 1`

:ANALysis:PMEAsure<x>:MEASure:HISTory:ABORt

Function Aborts the statistical processing of the history waveform of automated power measurement.

Syntax `:ANALysis:PMEAsure<x>:MEASure:HISTory:ABORt`
<x> = 1 or 2

Example `:ANALYSIS:PMEASURE1:MEASURE:HISTORY:ABORT`

:ANALysis:PMEAsure<x>:MEASure:HISTory:EXECute

Function Executes the statistical processing of the history waveform of automated power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:HISTory:EXECute
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:MEASURE:HISTORY:EXECUTE
```

:ANALysis:PMEAsure<x>:MEASure:MODE

Function Sets or queries the statistical processing type of automated power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:MODE {OFF|CONTinuous|CYCLe|HISTory}
:ANALysis:PMEAsure<x>:MEASure:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:MEASURE:MODE OFF
:ANALYSIS:PMEASURE1:MEASURE:MODE?
-> :ANALYSIS:PMEASURE1:MEASURE:MODE OFF
```

:ANALysis:PMEAsure<x>:MEASure:ZLINkage

Function Sets or queries the linkage (on/off) between the result numbers and zoom positions of the results of automated power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:ZLINkage {<Boolean>}
:ANALysis:PMEAsure<x>:MEASure:ZLINkage?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:MEASURE:ZLINKAGE ON
:ANALYSIS:PMEASURE1:MEASURE:ZLINKAGE?
-> :ANALYSIS:PMEASURE1:MEASURE:ZLINKAGE ON
```

:ANALysis:PMEAsure<x>:MODE

Function Sets or queries the on/off state of power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:MODE {<Boolean>}
:ANALysis:PMEAsure<x>:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:MODE ON
:ANALYSIS:PMEASURE1:MODE?
-> :ANALYSIS:PMEASURE1:MODE 1
```

:ANALysis:PMEAsure<x>:RANGe

Function Sets or queries the measurement source window.

Syntax
```
:ANALysis:PMEAsure<x>:RANGe {MAIN|Z1|Z2}
:ANALysis:PMEAsure<x>:RANGe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:RANGE MAIN
:ANALYSIS:PMEASURE1:RANGE?
-> :ANALYSIS:PMEASURE1:RANGE MAIN
```

:ANALysis:PMEAsure<x>:SETup?

Function Queries all power measurement input settings.

Syntax
```
:ANALysis:PMEAsure<x>:SETup?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:SETUP?
```

:ANALysis:PMEAsure<x>:SETup:ADESkew

Function Executes auto deskewing for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:ADESkew
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:SETUP:ADESKEW
```

:ANALysis:PMEAsure<x>:SETup:I?

Function Queries all current input channel settings for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:I?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:SETUP:I?
```

:ANALysis:PMEAsure<x>:SETup:I:DESKew

Function Sets or queries auto deskewing of the current input channel for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:I:DESKew {<time>}
:ANALysis:PMEAsure<x>:SETup:I:DESKew?
```
<x> = 1 or 2
<time> = –1000.00 ns to 1000.00 ns (in 10 ps steps)

Example
```
:ANALYSIS:PMEASURE1:SETUP:I:DESKEW 1NS
:ANALYSIS:PMEASURE1:SETUP:I:DESKEW?
-> :ANALYSIS:PMEASURE1:SETUP:I:DESKEW 1.000E-09
```

:ANALysis:PMEAsure<x>:SETup:I:PROBe

Function Queries all current-to-voltage conversion ratio settings of the probe connected to the current input channel for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:I:PRO
Be {C0_001|C0_002|C0_005|C0_01|C0_02
|C0_05|C0_1|C0_2|C0_5|C1|C2|C5|C10|C
20|C50|C100|C200|C500|C1000|C2000}
:ANALysis:PMEAsure<x>:SETup:I:PROBe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:SETUP:I:
PROBE C0_001
:ANALYSIS:PMEASURE1:SETUP:I:PROBE?
-> :ANALYSIS:PMEASURE1:SETUP:I:
PROBE C0_001
```

:ANALysis:PMEAsure<x>:SETup:RTRace

Function Sets or queries the deskewing source trace for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:RTRa
ce {I|U}
:ANALysis:PMEAsure<x>:SETup:RTRace?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:SETUP:RTRACE I
:ANALYSIS:PMEASURE1:SETUP:RTRACE?
-> :ANALYSIS:PMEASURE1:SETUP:RTRA
CE I
```

:ANALysis:PMEAsure<x>:SETup:U?

Function Queries all voltage input channel settings for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:U?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:SETUP:U?
```

:ANALysis:PMEAsure<x>:SETup:U:DESKew

Function Sets or queries auto deskewing of the voltage input channel for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:U:DESK
ew {<time>}
:ANALysis:PMEAsure<x>:SETup:U:DESK
ew?
```
<x> = 1 or 2

<time> = –1000.00 ns to 1000.00 ns (in 10 ps steps)

Example
```
:ANALYSIS:PMEASURE1:SETUP:U:DESK
EW 1NS
:ANALYSIS:PMEASURE1:SETUP:U:DESKEW?
-> :ANALYSIS:PMEASURE1:SETUP:U:DESK
EW 1.000E-09
```

:ANALysis:PMEAsure<x>:SETup:U:PROBe

Function Queries all voltage-to-voltage conversion ratio settings of the probe connected to the voltage input channel for power measurement.

Syntax
```
:ANALysis:PMEAsure<x>:SETup:U:PRO
Be {<NRf>}
:ANALysis:PMEAsure<x>:SETup:U:PROBe?
```
<x> = 1 or 2

<NRf> = See the Features Guide for this information.

Example
```
:ANALYSIS:PMEASURE1:SETUP:U:PROBE 1
:ANALYSIS:PMEASURE1:SETUP:U:PROBE?
-> :ANALYSIS:PMEASURE1:SETUP:U:
PROBE 1.000
```

:ANALysis:PMEAsure<x>:TRANge

Function Sets or queries the measurement time period.

Syntax
```
:ANALysis:PMEAsure<x>:TRANge {<NRf>,
<NRf>}
:ANALysis:PMEAsure<x>:TRANge?
```
<x> = 1 or 2

<NRf>, <NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:ANALYSIS:PMEASURE1:TRANGE 5,-5
:ANALYSIS:PMEASURE1:TRANGE?
-> :ANALYSIS:PMEASURE1:
TRANGE 5.0000000E+00,-5.0000000E+00
```

:ANALysis:PMEAsure<x>:UDPRoximal?

Function Queries all distal, mesial, and proximal settings.

Syntax
```
:ANALysis:PMEAsure<x>:UDPRoximal?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:UDPROXIMAL?
```

:ANALysis:PMEAsure<x>:UDPRoximal:MODE

Function Sets or queries the distal, mesial, and proximal point mode setting.

Syntax
```
:ANALysis:PMEAsure<x>:UDPRoximal:MO
DE {PERCent|UNIT}
:ANALysis:PMEAsure<x>:UDPRoximal:MO
DE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:UDPROXIMAL:MO
DE PERCENT
:ANALYSIS:PMEASURE1:UDPROXIMAL:MODE?
-> :ANALYSIS:PMEASURE1:UDPROXIMAL:MO
DE PERCENT
```

:ANALysis:PMEAsure<x>:UDPRoximal:PERCent

Function Sets or queries the distal, mesial, and proximal points as percentages.

Syntax
```
:ANALysis:PMEAsure<x>:UDPRoximal:PERCent {<NRf>,<NRf>,<NRf>}
:ANALysis:PMEAsure<x>:UDPRoximal:PERCent?
```
<x> = 1 or 2
<NRf>, <NRf>, <NRf> = 0 to 100 (%, in steps of 1)

Example
```
:ANALYSIS:PMEASURE1:UDPROXIMAL:PERCENT 10,50,90
:ANALYSIS:PMEASURE1:UDPROXIMAL:PERCENT?
-> :ANALYSIS:PMEASURE1:UDPROXIMAL:PERCENT 10,50,90
```

:ANALysis:PMEAsure<x>:UDPRoximal:UNIT

Function Sets or queries the distal, mesial, and proximal points as voltages.

Syntax
```
:ANALysis:PMEAsure<x>:UDPRoximal:UNIT {<NRf>,<NRf>,<NRf>}
:ANALysis:PMEAsure<x>:UDPRoximal:UNIT?
```
<x> = 1 or 2
<NRf>, <NRf>, <NRf> = See the Features Guide for this information.

Example
```
:ANALYSIS:PMEASURE1:UDPROXIMAL:UNIT -1,0,1
:ANALYSIS:PMEASURE1:UDPROXIMAL:UNIT?
-> :ANALYSIS:PMEASURE1:UDPROXIMAL:UNIT -1.0000000E+00,0.0000000E+00,1.0000000E+00
```

:ANALysis:PMEAsure<x>:UMEThod

Function Sets or queries the high and low points.

Syntax
```
:ANALysis:PMEAsure<x>:UMEThod {AUTO|MAXimum|HISTogram}
:ANALysis:PMEAsure<x>:UMEThod?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:UMETHOD AUTO
:ANALYSIS:PMEASURE1:UMETHOD?
-> :ANALYSIS:PMEASURE1:UMETHOD AUTO
```

:ANALysis:PMEAsure<x>:UNIT

Function Sets or queries the watthour unit.

Syntax
```
:ANALysis:PMEAsure<x>:UNIT {J|WH}
:ANALysis:PMEAsure<x>:UNIT?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PMEASURE1:UNIT J
:ANALYSIS:PMEASURE1:UNIT?
-> :ANALYSIS:PMEASURE1:UNIT J
```

:ANALysis:PMEAsure<x>:USER<y>?

Function Queries all automatic measurement settings for a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>?
```
<x> = 1 or 2
<y> = 1 to 4

Example
```
:ANALYSIS:PMEASURE:USER?
```

:ANALysis:PMEAsure<x>:USER<y>:COUNt?

Function Queries the statistical processing count of the automatically measured value of a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>:COUNt?
```
<x> = 1 or 2
<y> = 1 to 4

Example
```
:ANALYSIS:PMEASURE:USER:COUNT?
-> :ANALYSIS:PMEASURE:USER:COUNT 1
```

:ANALysis:PMEAsure<x>:USER<y>:DEFine

Function Sets or queries the expression for the automatically measured value of a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>:DEFine {<String>}
:ANALysis:PMEAsure<x>:USER<y>:DEFine?
```
<x> = 1 or 2
<y> = 1 to 4
<String> = Up to 128 characters

Example
```
:ANALYSIS:PMEASURE:USER:DEFINE "ABC"
:ANALYSIS:PMEASURE:USER:DEFINE?
-> :ANALYSIS:PMEASURE:USER:DEFINE "ABC"
```

:ANALysis:PMEAsure<x>:USER<y>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries a statistical value that is calculated on the automatically measured value of a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>:{MAXimum|MEAN|MINimum|SDEViation}?
```
<x> = 1 or 2
<y> = 1 to 4

Example
```
:ANALYSIS:PMEASURE:USER:MAXIMUM?
-> :ANALYSIS:PMEASURE:USER:MAXIMUM 0.0
```

Description If the statistical value is immeasurable, the instrument returns “NAN” (not a number).

:ANALysis:PMEAsure<x>:USER<y>:NAME

Function Sets or queries the name of a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>:NAME {<String>}
:ANALysis:PMEAsure<x>:USER<y>:NAME?
```
<x> = 1 or 2
<y> = 1 to 4
<String> = Up to 8 characters

Example
```
:ANALYSIS:PMEASURE:USER:NAME "ABC"
:ANALYSIS:PMEASURE:USER:NAME?
-> :ANALYSIS:PMEASURE:USER:NAME "ABC"
```

:ANALysis:PMEAsure<x>:USER<y>:STATe

Function Sets or queries the on/off status of automated measurement of a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>:STATe {<Boolean>}
:ANALysis:PMEAsure<x>:USER<y>:STATe?
```
<x> = 1 or 2
<y> = 1 to 4

Example
```
:ANALYSIS:PMEASURE:USER:STATE ON
:ANALYSIS:PMEASURE:USER:STATE?
-> :ANALYSIS:PMEASURE:USER:STATE 1
```

:ANALysis:PMEAsure<x>:USER<y>:UNIT

Function Sets or queries the unit of a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>:UNIT {<String>}
:ANALysis:PMEAsure<x>:USER<y>:UNIT?
```
<x> = 1 or 2
<y> = 1 to 4
<String> = Up to 4 characters

Example
```
:ANALYSIS:PMEASURE:USER:UNIT "ABC"
:ANALYSIS:PMEASURE:USER:UNIT?
-> :ANALYSIS:PMEASURE:USER:UNIT "ABC"
```

:ANALysis:PMEAsure<x>:USER<y>:VALue?

Function Queries the automatically measured value of a Calc item.

Syntax
```
:ANALysis:PMEAsure<x>:USER<y>:VALue? {<NRf>}
```
<x> = 1 or 2
<y> = 1 to 4
<NRf> = 1 to 100000

Example
```
:ANALYSIS:PMEASURE:USER:VALUE?
-> :ANALYSIS:PMEASURE:USER:VALUE 0.0
```

Description
- If the value is immeasurable, the instrument returns “NAN” (not a number).
- The <NRf> is used to specify which iteration of automated measurement to query the measured value from.
- If <NRf> is set to 1, the oldest measured value in the automated measurement memory is queried.
- If a measured value does not exist at the specified iteration, the instrument returns “NAN” (not a number).
- If <NRf> is omitted, the most recent measured value is queried.

:ANALysis:WAIT?

Function Waits for the completion of automated measurement with a timeout.

Syntax
```
:ANALysis:WAIT? {<NRf>}
```
<NRf> = 1 to 36000 (timeout period, 100 ms)

Example
```
:ANALYSIS:WAIT?
-> :ANALYSIS:WAIT 1
```

Description If the execution of automated measurement is completed within the timeout period, 0 is returned. If it is not completed or if automated measurement is not being performed, 1 is returned. Even if you set a long timeout period, 0 is returned as soon as the automated measurement is completed.

5.4 ASETup Group

:ASETup:EXECute

Function Executes auto setup.

Syntax `:ASETup:EXECute`

Example `:ASETUP:EXECUTE`

:ASETup:UNDO

Function Undoes auto setup.

Syntax `:ASETup:UNDO`

Example `:ASETUP:UNDO`

5.5 CALibrate Group

:CALibrate?

Function Queries all calibration settings.

Syntax `:CALibrate?`

:CALibrate[:EXECute]

Function Executes calibration.

Syntax `:CALibrate[:EXECute]`

Example `:CALIBRATE:EXECUTE`

:CALibrate:MODE

Function Sets or queries the auto calibration on/off state.

Syntax `:CALibrate:MODE {AUTO|OFF}`

Example
```
:CALIBRATE:MODE AUTO
:CALIBRATE:MODE?
-> :CALIBRATE:MODE AUTO
```

5.6 CHANnel Group

:CHANnel<x>?

Function Queries all vertical axis settings of a channel.

Syntax `:CHANnel<x>?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

:CHANnel<x>:ASCale[:EXECute]

Function Executes auto scaling of a channel.

Syntax `CHANnel<x>:ASCale[:EXECute]`
<x> = 1 to 4

Example `CHANNEL1:ASCALE:EXECUTE`

Description Auto scaling is possible on each channel.

- V/div
  Displayed so that the entire amplitude of the waveform can be seen without changing the vertical position.
- Offset
  When the input coupling is set to AC
  0 V
  When the input coupling is set to DC
  Center = (Max+Min)/2
- Trig Level
  DC offset position

:CHANnel<x>:BWIDth

Function Sets or queries the input filter of the specified channel.

Syntax `:CHANnel<x>:BWIDth {FULL|<Frequency>}`
`:CHANnel<x>:BWIDth?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
{<Frequency> }= See the Features Guide for this information.

Example
```
:CHANNEL1:BWIDTH FULL
:CHANNEL1:BWIDTH?
-> :CHANNEL1:BWIDTH FULL
```

:CHANnel<x>:COUPling

Function Sets or queries the input coupling of the specified channel.

Syntax `:CHANnel<x>:COUPling {AC|DC|DC50}`
`:CHANnel<x>:COUPling?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:COUPLING AC
:CHANNEL1:COUPLING?
-> :CHANNEL1:COUPLING AC
```

:CHANnel<x>:DESKew

Function Sets or queries the deskewing of the specified channel.

Syntax `:CHANnel<x>:DESKew {<Time>}`
`:CHANnel<x>:DESKew?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<time> = –1000.00 ns to 1000.00 ns (in 10 ps steps)

Example
```
:CHANNEL1:DESKEW 1NS
:CHANNEL1:DESKEW?
-> :CHANNEL1:DESKEW 1.000E-09
```

:CHANnel<x>:DISPlay

Function Sets or queries the display on/off state of the specified channel.

Syntax `:CHANnel<x>:DISPlay {<Boolean>}`
`:CHANnel<x>:DISPlay?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:DISPLAY ON
:CHANNEL1:DISPLAY?
-> :CHANNEL1:DISPLAY 1
```

:CHANnel<x>:INVert

Function Sets or queries the on/off state of the invert mode (waveform inversion display) .

Syntax `:CHANnel<x>:INVert {<Boolean>}`
`:CHANnel<x>:INVert?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:INVERT ON
:CHANNEL1:INVERT?
-> :CHANNEL1:INVERT 1
```

:CHANnel<x>:LABel?

Function Queries all settings related to the waveform label of the specified channel.

Syntax `:CHANnel<x>:LABel?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

:CHANnel<x>:LABel:DEFine

Function Sets or queries the waveform label of a channel.

Syntax `:CHANnel<x>:LABel[:DEFine] {<String>}`
`:CHANnel<x>:LABel[:DEFine]?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<String> = Up to 8 characters

Example
```
:CHANNEL1:LABEL:DEFINE "CH1"
:CHANNEL1:LABEL:DEFINE?
-> :CHANNEL1:LABEL:DEFINE "CH1"
```

Description You can only use the characters and symbols on the keyboard that appears on the instrument's screen.

:CHANnel<x>:LABel:DISPlay

Function Sets or queries the display on/off state of the specified channel's waveform label.

Syntax
```
:CHANnel<x>:LABel:
DISPlay {<Boolean>}
:CHANnel<x>:LABel:DISPlay?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:LABEL:DISPLAY ON
:CHANNEL1:LABEL:DISPLAY?
-> :CHANNEL1:LABEL:DISPLAY 1
```

:CHANnel<x>:LSCale?

Function Queries all the linear scaling settings of the specified channel.

Syntax
```
:CHANnel<x>:LSCale?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

:CHANnel<x>:LSCale:AVALue

Function Sets or queries the scaling coefficient A.

Syntax
```
:CHANnel<x>:LSCale:AVALue {<NRf>}
:CHANnel<x>:LSCale:AVALue?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:CHANNEL1:LSCALE:AVALUE 10
:CHANNEL1:LSCALE:AVALUE?
-> :CHANNEL1:LSCALE:
AVALUE 10.0000E+00
```

:CHANnel<x>:LSCale:BVALue

Function Sets or queries offset B.

Syntax
```
:CHANnel<x>:LSCale:BVALue {<NRf>}
:CHANnel<x>:LSCale:BVALue?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:CHANNEL1:LSCALE:BVALUE 10
:CHANNEL1:LSCALE:BVALUE?
-> :CHANNEL1:LSCALE:
BVALUE 10.0000E+00
```

:CHANnel<x>:LSCale:MODE

Function Sets or queries the on/off state of linear scaling.

Syntax
```
:CHANnel<x>:LSCale:MODE {<Boolean>}
:CHANnel<x>:LSCale:MODE?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:LSCALE:MODE ON
:CHANNEL1:LSCALE:MODE?
-> :CHANNEL1:LSCALE:MODE 1
```

:CHANnel<x>:LSCale:UNIT

Function Sets or queries a unit that is attached to linear scaling results.

Syntax
```
:CHANnel<x>:LSCale:UNIT {<String>}
:CHANnel<x>:LSCale:UNIT?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<String> = Up to 4 characters

Example
```
:CHANNEL1:LSCALE:UNIT "EU"
:CHANNEL1:LSCALE:UNIT?
-> :CHANNEL1:LSCALE:UNIT "EU"
```

Description You can only use the characters and symbols on the keyboard that appears on the instrument's screen. The scaled values are never affected by the unit.

:CHANnel<x>:OFFSet

Function Sets or queries the offset voltage of the specified channel.

Syntax
```
:CHANnel<x>:OFFSet {<Voltage>|<Current>
}
:CHANnel<x>:OFFSet?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<Voltage>, <Current> = See the Features Guide for this information.

Example
```
:CHANNEL1:OFFSET 0V
:CHANNEL1:OFFSET?
-> :CHANNEL1:OFFSET 0.000E+00
```

:CHANnel<x>:POSition

Function Sets or queries the vertical position of the specified channel.

Syntax
```
:CHANnel<x>:POSition {<NRf>}
:CHANnel<x>:POSition?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = –4 to 4 div

Example
```
:CHANNEL1:POSITION 1
:CHANNEL1:POSITION?
-> :CHANNEL1:POSITION 1.00E+00
```

:CHANnel<x>:PROBe?

Function Queries all settings related to the probe attenuation of the specified channel.

Syntax
```
:CHANnel<x>:PROBe?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

:CHANnel<x>:PROBe:DZCalibrate

Function Executes demagnetization and zero calibration of the specified channel.

Syntax
```
:CHANnel<x>:PROBe:DZCalibrate
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:PROBE:DZCALIBRATE
```

:CHANnel<x>:PROBe[:MODE]

Function Sets or queries the probe attenuation of the specified channel.

Syntax
```
:CHANnel<x>:PROBe[:MODE] {<NRf>|C0_001|C0_002|C0_005|C0_01|C0_02|C0_05|C0_1|C0_2|C0_5|C1|C2|C5|C10|C20|C50|C100|C200|C500|C1000|C2000}
:CHANnel<x>:PROBe[:MODE]?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:PROBE:MODE C0_001
:CHANNEL1:PROBE:MODE?
-> :CHANNEL1:PROBE:MODE C0_001
```

:CHANnel<x>:PROBe:PZCalibrate

Function Executes zero calibration of the specified channel's current probe.

Syntax
```
:CHANnel<x>:PROBe:PZCalibrate
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CHANNEL1:PROBE:PZCALIBRATE
```

:CHANnel<x>:VARiable

Function Sets or queries the specified channel's voltage scale in 0.01 V/div steps.

Syntax
```
:CHANnel<x>:VARiable {<Voltage>|<Current>}
:CHANnel<x>:VARiable?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<Voltage>, <Current> = See the User's Manual for this information.

Example
```
:CHANNEL1:VARIABLE 1V
:CHANNEL1:VARIABLE?
-> :CHANNEL1:VARIABLE 1.000E+00
```

:CHANnel<x>:VDIV

Function Sets or queries the specified channel's voltage scale (V/div).

Syntax
```
:CHANnel<x>:VDIV {<Voltage>|<Current>}
:CHANnel<x>:VDIV?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<Voltage>, <Current> = See the User's Manual for this information.

Example
```
:CHANNEL1:VDIV 2V
:CHANNEL1:VDIV?
-> :CHANNEL1:VDIV 2.000E+00
```

5.7 CLEar Group

:CLEar

Function Clears traces.

Syntax :CLEar

Example :CLEAR

5.8 COMMunicate Group

The commands in this group deal with communications. There are no front panel keys that correspond to the commands in this group.

:COMMunicate?

Function Queries all communication settings.

Syntax `:COMMunicate?`

:COMMunicate:HEADer

Function Sets or queries whether a header is added to the response to a query (example with header: “CHANNEL1:PROBE:MODE 10”; example without header: “10”).

Syntax
```
:COMMunicate:HEADer {<Boolean>}
:COMMunicate:HEADer?
```

Example
```
:COMMUNICATE:HEADER ON
:COMMUNICATE:HEADER?
-> :COMMUNICATE:HEADER 1
```

:COMMunicate:LOCKout

Function Sets/clears local lockout.

Syntax
```
:COMMunicate:LOCKout {<Boolean>}
:COMMunicate:LOCKout?
```

Example
```
:COMMUNICATE:LOCKOUT ON
:COMMUNICATE:LOCKOUT?
-> :COMMUNICATE:LOCKOUT 1
```

:COMMunicate:OPSE (Operation Pending Status Enable register)

Function Sets or queries the overlap command that is used by the *OPC, *OPC?, and *WAI commands.

Syntax
```
:COMMunicate:OPSE <Register>
:COMMunicate:OPSE?
```
<Register> = 0 to 65535 (see the figure for the :COMMunicate:WAIT? command)

Example
```
:COMMUNICATE:OPSE 65535
:COMMUNICATE:OPSE?
-> :COMMUNICATE:OPSE 2400
```

Description In the above example, all bits are set to 1 to make all overlap commands applicable. However, bits fixed to 0 are not set to 1, so the response to the query only indicates 1 for bits 5, 6, 8, and 11.

:COMMunicate:OPSR? (Operation Pending Status Register)

Function Queries the operation pending status register.

Syntax
```
:COMMunicate:OPSR?
```

Example
```
:COMMUNICATE:OPSR?
-> 0
```

Description For details on the operation pending status register, see the figure for the :COMMunicate:WAIT? command.

:COMMunicate:OVERlap

Function Sets or queries the commands that operate as overlap commands.

Syntax
```
:COMMunicate:OVERlap <Register>
:COMMunicate:OVERlap?
```
<Register> = 0 to 65535

Example
```
:COMMUNICATE:OVERLAP 65535
:COMMUNICATE:OVERLAP?
-> :COMMUNICATE:OVERLAP 2400
```

Description
- In the above example, all bits are set to 1 to make all overlap commands applicable. However, bits fixed to 0 are not set to 1, so the response to the query only indicates 1 for bits 5, 6, 8, and 11.
- For information about how to synchronize a program using COMMunicate:OVERlap, see page 4-8.
- In the above example, bits 5, 6, 8, and 11 are set to 1 to make all overlap commands applicable (see the figure for the :COMMunicate:WAIT? command).

:COMMunicate:REMote

Function Sets the instrument to remote or local mode. On is remote mode.

Syntax
```
:COMMunicate:REMote {<Boolean>}
:COMMunicate:REMote?
```

Example
```
:COMMUNICATE:REMOTE ON
:COMMUNICATE:REMOTE?
-> :COMMUNICATE:REMOTE 1
```

:COMMunicate:VERBose

Function Sets or queries whether the response to a query is returned fully spelled out (e.g., CHANNEL1:PROBE:MODE 10) or in its abbreviated form (e.g., CHAN:PROB 10).

Syntax
```
:COMMunicate:VERBose {<Boolean>}
:COMMunicate:VERBose?
```

Example
```
:COMMUNICATE:VERBOSE ON
:COMMUNICATE:VERBOSE?
-> :COMMUNICATE:VERBOSE 1
```

:COMMunicate:WAIT

Function Waits for a specified extended event to occur.

Syntax
```
:COMMunicate:WAIT <Register>
```
<Register> = 0 to 65535 (Extended event register. For more information, see page 6-5.)

Example
```
:COMMUNICATE:WAIT 65535
```

Description For information about how to synchronize a program using COMMunicate:WAIT, see page 4-9.

:COMMunicate:WAIT?

Function Creates the response that is returned when a specified extended event occurs.

Syntax
```
:COMMunicate:WAIT? <Register>
```
<Register> = 0 to 65535 (Extended event register. For more information, see page 6-5.)

Example
```
:COMMUNICATE:WAIT? 65535
-> 1
```

Description Operation pending status register and overlap enable register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	SCH	0	0	HST	0	ACS	PRN	0	0	0	0	0

When bit 5 (PRN) = 1:
Built-in printer operation not complete
When bit 6 (ACS) = 1:
Media access not complete
When bit 8 (HST) = 1:
History search execution not complete
When bit 11 (SCH) = 1:
Edge/pattern search execution not complete

5.9 CURSor Group

:CURSor?
Function Queries all cursor measurement settings.
Syntax :CURSor?

:CURSor[:TY]?
Function Queries all cursor settings.
Syntax :CURSor[:TY]?

:CURSor[:TY]:DEGRee?
Function Queries all angle cursor settings.
Syntax :CURSor[:TY]:DEGRee?

:CURSor[:TY]:DEGRee:ALL
Function Collectively turns on or off the measured values of the angle cursor on the VT display.
Syntax :CURSor:TY:DEGRee:ALL {<Boolean>}
Example :CURSOR:TY:DEGREE:ALL ON

:CURSor[:TY]:DEGRee:D<x>?
Function Queries all angle cursor's angle settings.
Syntax :CURSor[:TY]:DEGRee:D<x>?
<x> = 1 or 2

:CURSor[:TY]:DEGRee:D<x>:STATe
Function Sets or queries the on/off state of the angle cursor's angle.
Syntax :CURSor[:TY]:DEGRee:D<x>:
STATe {<Boolean>}
:CURSor[:TY]:DEGRee:D<x>:STATe?
<x> = 1 or 2
Example :CURSOR:TY:DEGREE:D1:STATE ON
:CURSOR:TY:DEGREE:D1:STATE?
-> :CURSOR:TY:DEGREE:D1:STATE 1

:CURSor[:TY]:DEGRee:D<x>:VALue?
Function Queries the the angle cursor's angle.
Syntax :CURSor[:TY]:DEGRee:D<x>:VALue?
<x> = 1 or 2
Example :CURSOR:TY:DEGREE:D1:VALUE?
-> :CURSOR:TY:DEGREE:D1:
VALUE -120.00000E+00

:CURSor[:TY]:DEGRee:DD?
Function Queries all settings related to the angle difference between the angle cursors.
Syntax :CURSor[:TY]:DEGRee:DD?

:CURSor[:TY]:DEGRee:DD:STATe
Function Sets or queries the on/off state of the angle difference ΔD value between the angle cursors.
Syntax :CURSor[:TY]:DEGRee:DD:
STATe {<Boolean>}
:CURSor[:TY]:DEGRee:DD:STATe?
Example :CURSOR:TY:DEGREE:DD:STATE ON
:CURSOR:TY:DEGREE:DD:STATE?
-> :CURSOR:TY:DEGREE:DD:STATE 1

:CURSor[:TY]:DEGRee:DD:VALue?
Function Queries the angle difference ΔD value between the angle cursors.
Syntax :CURSor[:TY]:DEGRee:DD:VALue?
Example :CURSOR:TY:DEGREE:DD:VALUE?
-> :CURSOR:TY:DEGREE:DD:
VALUE 180.00000E+00

:CURSor[:TY]:DEGRee:DV?
Function Queries all settings related to the ΔV value between the angle cursors.
Syntax :CURSor[:TY]:DEGRee:DV?

:CURSor[:TY]:DEGRee:DV:STATe
Function Sets or queries the on/off state of the ΔV value between the angle cursors.
Syntax :CURSor[:TY]:DEGRee:DV:
STATe {<Boolean>}
:CURSor[:TY]:DEGRee:DV:STATe?
Example :CURSOR:TY:DEGREE:DV:STATE ON
:CURSOR:TY:DEGREE:DV:STATE?
-> :CURSOR:TY:DEGREE:DV:STATE 1

:CURSor[:TY]:DEGRee:DV:VALue?
Function Queries the ΔV value between the angle cursors.
Syntax :CURSor[:TY]:DEGRee:DV:VALue?
Example :CURSOR:TY:DEGREE:DV:VALUE?
-> :CURSOR:TY:DEGREE:DV:
VALUE 6.2500000E-03

:CURSor[:TY]:DEGRee:JUMP

Function Moves the angle cursor on the VT display on to the specified zoom waveform.

Syntax `:CURSor[:TY]:DEGRee:JUMP {C1_Z1|C1_Z2|C2_Z1|C2_Z2}`

Example `:CURSOR:TY:DEGREE:JUMP C1_Z1`

Description This command moves the cursor to the center position of the zoom waveform. C1 and C2 are used to indicate Cursor1 and Cursor2.

:CURSor[:TY]:DEGRee:POSition<x>

Function Sets or queries the angle cursor position.

Syntax `:CURSor[:TY]:DEGRee:POSition<x> {<NRf>}`
`:CURSor[:TY]:DEGRee:POSition<x>?`
<x> = 1 or 2
<NRf> = –5 div to 5 div (in steps of 10 divisions/display record length)

Example
```
:CURSOR:TY:DEGREE:POSITION1 2
:CURSOR:TY:DEGREE:POSITION1?
-> :CURSOR:TY:DEGREE:POSITION1 2.0000000
```

:CURSor[:TY]:DEGRee:REFerence<x>

Function Sets or queries the start point (Reference1) or end point (Reference2) of the reference angle.

Syntax `:CURSor[:TY]:DEGRee:REFerence<x> {<NRf>}`
`:CURSor[:TY]:DEGRee:REFerence<x>?`
<x> = 1 or 2
<NRf> = –5 div to 5 div (in steps of 10 divisions/display record length)

Example
```
:CURSOR:TY:DEGREE:REFERENCE1 -1
:CURSOR:TY:DEGREE:REFERENCE1?
-> :CURSOR:TY:DEGREE:REFERENCE1 -1.0000000
```

:CURSor[:TY]:DEGRee:RVALue

Function Sets or queries the reference angle.

Syntax `:CURSor[:TY]:DEGRee:RVALue {<NRf>}`
`:CURSor[:TY]:DEGRee:RVALue?`
<NRf> = 1 to 720

Example
```
:CURSOR:TY:DEGREE:RVALUE 180
:CURSOR:TY:DEGREE:RVALUE?
-> :CURSOR:TY:DEGREE:RVALUE 180
```

:CURSor[:TY]:DEGRee:TRACe

Function Sets or queries the source waveform of the angle cursor.

Syntax `:CURSor[:TY]:DEGRee:TRACe {<NRf>|ALL|MATH<x>}`
`:CURSor[:TY]:DEGRee:TRACe?`
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CURSOR:TY:DEGREE:TRACE 1
:CURSOR:TY:DEGREE:TRACE?
-> :CURSOR:TY:DEGREE:TRACE 1
```

:CURSor[:TY]:DEGRee:UNIT

Function Sets or queries the angle cursor unit.

Syntax `:CURSor[:TY]:DEGRee:UNIT {<String>}`
`:CURSor[:TY]:DEGRee:UNIT?`
<String> = Up to 4 characters

Example
```
:CURSOR:TY:DEGREE:UNIT "DEG"
:CURSOR:TY:DEGREE:UNIT?
-> :CURSOR:TY:DEGREE:UNIT "DEG"
```

:CURSor[:TY]:DEGRee:V<x>?

Function Queries all angle cursor's voltage settings.

Syntax `:CURSor[:TY]:DEGRee:V<x>?`
<x> = 1 or 2

:CURSor[:TY]:DEGRee:V<x>:STATe

Function Sets or queries the on/off state of the angle cursor's voltage.

Syntax `:CURSor[:TY]:DEGRee:V<x>:STATe {<Boolean>}`
`:CURSor[:TY]:DEGRee:V<x>:STATe?`
<x> = 1 or 2

Example
```
:CURSOR:TY:DEGREE:V1:STATE ON
:CURSOR:TY:DEGREE:V1:STATE?
-> :CURSOR:TY:DEGREE:V1:STATE 1
```

:CURSor[:TY]:DEGRee:V<x>:VALue?

Function Queries the angle cursor's voltage.

Syntax `:CURSor[:TY]:DEGRee:V<x>:VALue?`
<x> = 1 or 2

Example
```
:CURSOR:TY:DEGREE:V1:VALUE?
-> :CURSOR:TY:DEGREE:V1:VALUE 10.000000E-03
```

:CURSor[:TY]:HORizontal?

Function Queries all ΔV cursor settings.

Syntax `:CURSor[:TY]:HORizontal?`

:CURSor[:TY]:HORizontal:ALL

Function Collectively turns on or off the measured values of the ΔV cursor on the VT display.

Syntax
```
:CURSor:TY:HORizontal:
ALL {<Boolean>}
```
Example
```
:CURSOR:TY:HORIZONTAL:ALL ON
```

:CURSor[:TY]:HORizontal:DV?

Function Queries all settings related to the vertical value between the ΔV cursors.

Syntax
```
:CURSor[:TY]:HORizontal:DV?
```

:CURSor[:TY]:HORizontal:DV:STATe

Function Sets or queries the on/off state of the vertical value between the ΔV cursors.

Syntax
```
:CURSor[:TY]:HORizontal:DV:STA
Te {<Boolean>}
:CURSor[:TY]:HORizontal:DV:STATe?
```
Example
```
:CURSOR:TY:HORIZONTAL:DV:STATE ON
:CURSOR:TY:HORIZONTAL:DV:STATE?
-> :CURSOR:TY:HORIZONTAL:DV:STATE 1
```

:CURSor[:TY]:HORizontal:DV:VALue?

Function Queries the vertical value between the ΔV cursors.

Syntax
```
:CURSor[:TY]:HORizontal:DV:VALue?
```
Example
```
:CURSOR:TY:HORIZONTAL:DV:VALUE?
-> :CURSOR:TY:HORIZONTAL:DV:
VALUE 3.0000000E+00
```
Description If linear scaling is set to ON, the scaled value will be queried.

:CURSor[:TY]:HORizontal:POSition<x>

Function Sets or queries the ΔV cursor position.

Syntax
```
:CURSor[:TY]:HORizontal:POSition
<x> {<NRf>}
:CURSor[:TY]:HORizontal:POSition<x>?
```
<x> = 1 or 2
<NRf> = –4 to 4 (in 1/ 100 steps)

Example
```
:CURSOR:TY:HORIZONTAL:POSITION1 -4
:CURSOR:TY:HORIZONTAL:POSITION1?
-> :CURSOR:TY:HORIZONTAL:
POSITION1 -4
```
Description <NRf> is significant to two decimal places.

:CURSor[:TY]:HORizontal:TRACe

Function Sets or queries the source waveform of the ΔV cursor.

Syntax
```
:CURSor[:TY]:HORizontal:
TRACe {<NRf>|MATH<x>}
:CURSor[:TY]:HORizontal:TRACe?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CURSOR:TY:HORIZONTAL:TRACE 1
:CURSOR:TY:HORIZONTAL:TRACE?
-> :CURSOR:TY:HORIZONTAL:TRACE 1
```

:CURSor[:TY]:HORizontal:V<x>?

Function Queries all settings related to the ΔV cursor's vertical axis.

Syntax
```
:CURSor[:TY]:HORizontal:V<x>?
```
<x> = 1 or 2

:CURSor[:TY]:HORizontal:V<x>:STATe

Function Sets or queries the on/off state of the ΔV cursor's vertical value.

Syntax
```
:CURSor[:TY]:HORizontal:V<x>:STA
Te {<Boolean>}
:CURSor[:TY]:HORizontal:V<x>:STATe?
```
<x> = 1 or 2

Example
```
:CURSOR:TY:HORIZONTAL:V1:STATE ON
:CURSOR:TY:HORIZONTAL:V1:STATE?
-> :CURSOR:TY:HORIZONTAL:V1:STATE 1
```

:CURSor[:TY]:HORizontal:V<x>:VALue?

Function Queries the ΔV cursor's vertical value.

Syntax
```
:CURSor[:TY]:HORizontal:V<x>:VALue?
```
<x> = 1 or 2

Example
```
:CURSOR:TY:HORIZONTAL:V1:VALUE?
-> :CURSOR:TY:HORIZONTAL:V1:
VALUE -1.5000000E+00
```
Description If linear scaling is set to ON, the scaled value will be queried.

:CURSor[:TY]:MARKer?

Function Queries all marker cursor settings.

Syntax
```
:CURSor[:TY]:MARKer?
```

:CURSor[:TY]:MARKer:FORM

Function Sets or queries the display format of the marker cursor.

Syntax
```
:CURSor[:TY]:MARKer:FORM {LINE|MARK}
:CURSor[:TY]:MARKer:FORM?
```

Example
```
:CURSOR:TY:MARKER:FORM LINE
:CURSOR:TY:MARKER:FORM?
-> :CURSor[:TY]:MARKER:FORM LINE
```

:CURSor[:TY]:MARKer:M<x>?

Function Queries all settings related to the specified marker cursor.

Syntax `:CURSor[:TY]:MARKer:M<x>?`
<x> = 1 to 4

:CURSor[:TY]:MARKer:M<x>:ALL

Function Collectively turns on or off the measured values of the marker cursor on the VT display.

Syntax `:CURSor:TY:MARKer:M<x>:ALL {<Boolean>}`
<x> = 1 to 4

Example `:CURSOR:TY:MARKER:M1:ALL ON`

:CURSor[:TY]:MARKer:M<x>:DT<y>?

Function Queries all settings related to the time value between the marker cursors.

Syntax `:CURSor[:TY]:MARKer:M<x>:DT<y>?`
<x> = 1 to 4
<y> = 1 to 4

:CURSor[:TY]:MARKer:M<x>:DT<y>:STATe

Function Sets or queries the on/off state of the time value between the marker cursors.

Syntax `:CURSor[:TY]:MARKer:M<x>:DT<y>:STATe {<Boolean>}`
`:CURSor[:TY]:MARKer:M<x>:DT<y>:STATe?`
<x> = 1 to 4
<y> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M:DT1:STATE ON
:CURSOR:TY:MARKER:M:DT1:STATE?
-> :CURSOR:TY:MARKER:M:DT1:STATE 1
```

:CURSor[:TY]:MARKer:M<x>:DT<y>:VALue?

Function Queries the time value between the marker cursors.

Syntax `:CURSor[:TY]:MARKer:M<x>:DT<y>:VALue?`
<x> = 1 to 4
<y> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M:DT1:VALUE?
-> :CURSOR:TY:MARKER:M:DT1:
VALUE 0.0000000E+00
```

:CURSor[:TY]:MARKer:M<x>:DV<y>?

Function Queries all settings related to the vertical value between the marker cursors.

Syntax `:CURSor[:TY]:MARKer:M<x>:DV<y>?`
<x> = 1 to 4
<y> = 1 to 4

:CURSor[:TY]:MARKer:M<x>:DV<y>:STATe

Function Sets or queries the on/off state of the vertical value between the marker cursors.

Syntax `:CURSor[:TY]:MARKer:M<x>:DV<y>:STATe {<Boolean>}`
`:CURSor[:TY]:MARKer:M<x>:DV<y>:STATe?`
<x> = 1 to 4
<y> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M:DV1:STATE ON
:CURSOR:TY:MARKER:M:DV1:STATE?
-> :CURSOR:TY:MARKER:M:DV1:STATE 1
```

:CURSor[:TY]:MARKer:M<x>:DV<y>:VALue?

Function Queries the vertical value between the marker cursors.

Syntax `:CURSor[:TY]:MARKer:M<x>:DV<y>:VALue?`
<x> = 1 to 4
<y> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M:DV1:VALUE?
-> :CURSOR:TY:MARKER:M:DV1:
VALUE 500.00000E-03
```

Description If linear scaling is set to ON, the scaled value will be queried.

:CURSor[:TY]:MARKer:M<x>:JUMP

Function Moves the marker cursor on the VT display on to the specified zoom waveform.

Syntax `:CURSor[:TY]:MARKer:M<x>:JUMP {Z1|Z2}`
<x> = 1 to 4

Example `:CURSOR:TY:MARKER:M1:JUMP Z1`

Description This command moves the cursor to the center position of the zoom waveform.

:CURSor[:TY]:MARKer:M<x>:POSition

Function Sets or queries the marker cursor's time value.

Syntax `:CURSor[:TY]:MARKer:M<x>:POSition {<NRf>}`
`:CURSor[:TY]:MARKer:M<x>:POSition?`
<x> = 1 to 4
<NRf> = –5 div to 5 div (in steps of 10 divisions/display record length)

Example
```
:CURSOR:TY:MARKER:M1:POSITION -1
:CURSOR:TY:MARKER:M1:POSITION?
-> :CURSOR:TY:MARKER:M1:
POSITION -1.0000000
```

:CURSor[:TY]:MARKer:M<x>:T?

Function Queries all settings related to the marker cursor's time value.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:T?
```
<x> = 1 to 4

:CURSor[:TY]:MARKer:M<x>:T:STATe

Function Sets or queries the on/off state of the marker cursor's time value.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:T:STA
Te {<Boolean>}
:CURSor[:TY]:MARKer:M<x>:T:STATe?
```
<x> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M1:T:STATE ON
:CURSOR:TY:MARKER:M1:T:STATE?
-> :CURSOR:TY:MARKER:M1:T:STATE 1
```

:CURSor[:TY]:MARKer:M<x>:T:VALue?

Function Queries the marker cursor's time value.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:T:VALue?
```
<x> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M1:T:VALUE?
-> :CURSOR:TY:MARKER:M1:T:
VALUE -4.5000E-03
```

:CURSor[:TY]:MARKer:M<x>:TRACe

Function Sets or queries the source waveform of the marker cursor.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:
TRACe {<NRf>|MATH<y>|OFF}
:CURSor[:TY]:MARKer:M<x>:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CURSOR:TY:MARKER:M1:TRACE 1
:CURSOR:TY:MARKER:M1:TRACE?
-> :CURSOR:TY:MARKER:M1:TRACE 1
```

:CURSor[:TY]:MARKer:M<x>:V?

Function Queries all settings related to the marker cursor's vertical value.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:V?
```
<x> = 1 to 4

:CURSor[:TY]:MARKer:M<x>:V:STATe

Function Sets or queries the on/off state of the marker cursor's vertical value.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:V:STA
Te {<Boolean>}
:CURSor[:TY]:MARKer:M<x>:V:STATe?
```
<x> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M1:V:STATE ON
:CURSOR:TY:MARKER:M1:V:STATE?
-> :CURSOR:TY:MARKER:M1:V:STATE 1
```

:CURSor[:TY]:MARKer:M<x>:V:VALue?

Function Queries the marker cursor's vertical value.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:V:VALue?
```
<x> = 1 to 4

Example
```
:CURSOR:TY:MARKER:M1:V:VALUE?
-> :CURSOR:TY:MARKER:M1:V:
VALUE 1.5000E-03
```

Description If linear scaling is set to ON, the scaled value will be queried.

:CURSor[:TY]:TYPE

Function Sets or queries the cursor type.

Syntax
```
:CURSor[:TY]:TYPE {OFF|HORizontal|HA
Vertical|MARKer|VERTical|DEGRee}
:CURSor[:TY]:TYPE?
```

Example
```
:CURSOR:TY:TYPE HORIZONAL
:CURSOR:TY:TYPE?
-> :CURSOR:TY:TYPE HORIZONTAL
```

:CURSor[:TY]:VERTical?

Function Queries all ΔT cursor settings.

Syntax
```
:CURSor[:TY]:VERTical?
```

:CURSor[:TY]:VERTical:ALL

Function Collectively turns on or off the measured values of the ΔT cursor on the VT display.

Syntax
```
:CURSor:TY:VERTical:ALL {<Boolean>}
```

Example
```
:CURSOR:TY:VERTICAL:ALL ON
```

:CURSor[:TY]:VERTical:DT?

Function Queries all settings related to the time value between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DT?
```

:CURSor[:TY]:VERTical:DT:STATe

Function Sets or queries the on/off state of the time value between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DT:
STATe {<Boolean>}
:CURSor[:TY]:VERTical:DT:STATe?
```

Example
```
:CURSOR:TY:VERTICAL:DT:STATE ON
:CURSOR:TY:VERTICAL:DT:STATE?
-> :CURSOR:TY:VERTICAL:DT:STATE 1
```

:CURSor[:TY]:VERTical:DT:VALue?

Function Queries the time value between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DT:VALue?
```

Example
```
:CURSOR:TY:VERTICAL:DT:VALUE?
-> :CURSOR:TY:VERTICAL:DT:
VALUE 2.50E-06
```

:CURSor[:TY]:VERTical:DV?

Function Queries all settings related to the vertical value between the ΔT cursors.

Syntax `:CURSor[:TY]:VERTical:DV?`

:CURSor[:TY]:VERTical:DV:STATe

Function Sets or queries the on/off state of the vertical value between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DV:
STATe {<Boolean>}
:CURSor[:TY]:VERTical:DV:STATe?
```

Example
```
:CURSOR:TY:VERTICAL:DV:STATE ON
:CURSOR:TY:VERTICAL:DV:STATE?
-> :CURSOR:TY:VERTICAL:DV:STATE 1
```

:CURSor[:TY]:VERTical:DV:VALue?

Function Queries the vertical value between the ΔT cursors.

Syntax `:CURSor[:TY]:VERTical:DV:VALue?`

Example
```
:CURSOR:TY:VERTICAL:DV:VALUE?
-> :CURSOR:TY:VERTICAL:DV:
VALUE 1.50E+03
```

Description If linear scaling is set to ON, the scaled value will be queried.

:CURSor[:TY]:VERTical:JUMP

Function Moves the vertical cursor on the VT display on to the specified zoom waveform.

Syntax `:CURSor[:TY]:VERTical:JUMP {C1_Z1|C1_Z2|C2_Z1|C2_Z2}`

Example `:CURSOR:TY:VERTICAL:JUMP C1_Z1`

Description This command moves the cursor to the center position of the zoom waveform. C1 and C2 are used to indicate V cursors Cursor1 and Cursor2.

:CURSor[:TY]:VERTical:PERDt?

Function Queries all settings related to the 1/ΔT value between the ΔT cursors.

Syntax `:CURSor[:TY]:VERTical:PERDt?`

:CURSor[:TY]:VERTical:PERDt:STATe

Function Sets or queries the on/off state of the 1/ΔT value between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:PERDt:
STATe {<Boolean>}
:CURSor[:TY]:VERTical:PERDt:STATe?
```

Example
```
:CURSOR:TY:VERTICAL:PERDT:STATE ON
:CURSOR:TY:VERTICAL:PERDT:STATE?
-> :CURSOR:TY:VERTICAL:PERDT:STATE 1
```

:CURSor[:TY]:VERTical:PERDt:VALue?

Function Queries the 1/ΔT value between the ΔT cursors.

Syntax `:CURSor[:TY]:VERTical:PERDt:VALue?`

Example
```
:CURSOR:TY:VERTICAL:PERDT:VALUE?
-> :CURSOR:TY:VERTICAL:PERDT:
VALUE 2.50E+06
```

:CURSor[:TY]:VERTical:POSition<x>

Function Sets or queries the ΔT cursor position.

Syntax
```
:CURSor[:TY]:VERTical:POSition
<x> {<NRf>}
:CURSor[:TY]:VERTical:POSition<x>?
```
<x> = 1 or 2
<NRf> = –5 div to 5 div (in steps of 10 divisions/display record length)

Example
```
:CURSOR:TY:VERTICAL:POSITION1 2
:CURSOR:TY:VERTICAL:POSITION1?
-> :CURSOR:TY:VERTICAL:
POSITION1 2.00E+00
```

:CURSor[:TY]:VERTical:T<x>?

Function Queries all settings related to the ΔT cursor's time value.

Syntax `:CURSor[:TY]:VERTical:T<x>?`
<x> = 1 or 2

:CURSor[:TY]:VERTical:T<x>:STATe

Function Sets or queries the on/off state of the ΔT cursor's time value.

Syntax
```
:CURSor[:TY]:VERTical:T<x>:STA
Te {<Boolean>}
:CURSor[:TY]:VERTical:T<x>:STATe?
```
<x> = 1 or 2

Example
```
:CURSOR:TY:VERTICAL:T1:STATE ON
:CURSOR:TY:VERTICAL:T1:STATE?
-> :CURSOR:TY:VERTICAL:T1:STATE 1
```

:CURSor[:TY]:VERTical:T<x>:VALue?

Function Queries the ΔT cursor's time value.

Syntax `:CURSor[:TY]:VERTical:T<x>:VALue?`
<x> = 1 or 2

Example
```
:CURSOR:TY:VERTICAL:T1:VALUE?
-> :CURSOR:TY:VERTICAL:T1:
VALUE -2.50E-06
```

:CURSor[:TY]:VERTical:TRACe

Function Sets or queries the source waveform of the ΔT cursor.

Syntax
```
:CURSor[:TY]:VERTical:TRACe {<NRf>|
ALL|MATH<x>}
:CURSor[:TY]:VERTical:TRACe?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:CURSOR:TY:VERTICAL:TRACE 1
:CURSOR:TY:VERTICAL:TRACE?
-> :CURSOR:TY:VERTICAL:TRACE 1
```

:CURSor[:TY]:VERTical:V<x>?

Function Queries all settings related to the ΔT cursor's vertical value.

Syntax `:CURSor[:TY]:VERTical:V<x>?`
<x> = 1 or 2

:CURSor[:TY]:VERTical:V<x>:STATe

Function Sets or queries the on/off state of the ΔT cursor's vertical value.

Syntax `:CURSor[:TY]:VERTical:V<x>:STATe {<Boolean>}`
`:CURSor[:TY]:VERTical:V<x>:STATe?`
<x> = 1 or 2

Example
```
:CURSOR:TY:VERTICAL:V1:STATE ON
:CURSOR:TY:VERTICAL:V1:STATE?
-> :CURSOR:TY:VERTICAL:V1:STATE 1
```

:CURSor[:TY]:VERTical:V<x>:VALue?

Function Queries the ΔT cursor's vertical value.

Syntax `:CURSor[:TY]:VERTical:V<x>:VALue?`
<x> = 1 or 2

Example
```
:CURSOR:TY:VERTICAL:V1:VALUE?
-> :CURSOR:TY:VERTICAL:V1:
VALUE 2.50E+03
```

5.10 DISPlay Group

:DISPlay?

Function Queries all display settings.

Syntax `:DISPlay?`

:DISPlay:ACCumulate?

Function Queries all accumulated waveform display settings.

Syntax `:DISPlay:ACCumulate?`

:DISPlay:ACCumulate:MODE

Function Sets or queries the accumulation mode.

Syntax `:DISPlay:ACCumulate:MODE {COLor|OFF|PERSistence}`
`:DISPlay:ACCumulate:MODE?`

Example `:DISPLAY:ACCUMULATE:MODE COLOR`
`:DISPLAY:ACCUMULATE:MODE?`
`-> :DISPLAY:ACCUMULATE:MODE COLOR`

:DISPlay:ACCumulate:PERSistence

Function Sets or queries the accumulation time.

Syntax `:DISPlay:ACCumulate:PERSistence {<Time>|INFinity}`
`:DISPlay:ACCumulate:PERSistence?`
<Time> = 100ms to 100s

Example `:DISPLAY:ACCUMULATE:PERSISTENCE 100ms`
`:DISPLAY:ACCUMULATE:PERSISTENCE?`
`-> :DISPLAY:ACCUMULATE:PERSISTENCE 100ms`

:DISPlay:COLor?

Function Queries all waveform color settings.

Syntax `:DISPlay:COLor?`

:DISPlay:COLor:{CHANnel<x>|MATH<x>}

Function Sets or queries a waveform color.

Syntax `:DISPlay:COLor:{CHANnel<x>|MATH<x>} {BLUE|BGReen|CYAN|DBLue|GRAY|GREen|LBLue|LGReen|MAGenta|MGReen|ORANge|PINK|PURPle|RED|SPINk|YELLow}`
`:DISPlay:COLor:{CHANnel<x>|MATH<x>}?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (2 on 2-channel models)

Example `:DISPLAY:COLOR:CHANNEL1 BLUE`
`:DISPLAY:COLOR:CHANNEL1?`
`-> :DISPLAY:COLOR:CHANNEL1 BLUE`

:DISPlay:COLor:LSTate

Function Sets or queries the logic waveform state display color.

Syntax `:DISPlay:COLor:LSTate {BLUE|BGReen|CYAN|DBLue|GRAY|GREen|LBLue|LGReen|MAGenta|MGReen|ORANge|PINK|PURPle|RED|SPINk|YELLow}`
`:DISPlay:COLor:LSTate?`

Example `:DISPLAY:COLOR:LSTATE BLUE`
`:DISPLAY:COLOR:LSTATE?`
`-> :DISPLAY:COLOR:LSTATE BLUE`

:DISPlay:COLor:SERialbus<x>?

Function Queries the color settings of all trends of the specified serial bus.

Syntax `:DISPlay:COLor:SERialbus<x>?`
<x> = 1 to 4

Description This command can be used on models with the serial bus option that supports trend display.

:DISPlay:COLor:SERialbus<x>:TRENd<y>

Function Sets or queries the serial bus trend color.

Syntax `:DISPlay:COLor:SERialbus<x>:TRENd<y> {BLUE|BGReen|CYAN|DBLue|GRAY|GREen|LBLue|LGReen|MAGenta|MGReen|ORANge|PINK|PURPle|RED|SPINk|YELLow}`
`:DISPlay:COLor:SERialbus<x>:TRENd<y>?`
<x> = 1 to 4
<y> = 1 to 4

Example `:DISPLAY:COLOR:SERIALBUS1:TREND1 BLUE`
`:DISPLAY:COLOR:SERIALBUS1:TREND1?`
`-> :DISPLAY:COLOR:SERIALBUS1:TREND1 BLUE`

Description This command can be used on models with the serial bus option that supports trend display.

:DISPlay:FORMat

Function Sets or queries the display format.

Syntax `:DISPlay:FORMat {AUTO|SINGle|DUAL|TRIad|QUAD|HEXa|OCTal}`
`:DISPlay:FORMat?`

Example `:DISPLAY:FORMAT AUTO`
`:DISPLAY:FORMAT?`
`-> :DISPLAY:FORMAT AUTO`

Description {HEXa|OCTal} is not available on 2-channel models.

:DISPlay:FGRid

Function Sets or queries whether the fine grid is displayed.

Syntax
```
:DISPlay:FGRid {<Boolean>}
:DISPlay:FGRid?
```

Example
```
:DISPLAY:FGRID ON
:DISPLAY:FGRID?
-> :DISPLAY:FGRID 1
```

:DISPlay:GRATicule

Function Sets or queries the graticule (grid).

Syntax
```
:DISPlay:GRATicule {CROSshair|FRAMe|
GRID|line}
:DISPlay:GRATicule?
```

Example
```
:DISPLAY:GRATICULE CROSSHAIR
:DISPLAY:GRATICULE?
-> :DISPLAY:GRATICULE CROSSHAIR
```

:DISPlay:INTENsity?

Function Queries all settings related to the intensity of display items.

Syntax
```
:DISPlay:INTENsity?
```

:DISPlay:INTENsity[:WAVeform]

Function Sets or queries the waveform intensity.

Syntax
```
:DISPlay:INTENsity[:
WAVeform] {<NRf>}
:DISPlay:INTENsity[:WAVeform]?
```
<NRf> = 1 to 64

Example
```
:DISPLAY:INTENSITY:WAVEFORM 10
:DISPLAY:INTENSITY:WAVEFORM?
-> :DISPLAY:INTENSITY:WAVEFORM 10
```

:DISPlay:INTENsity:{CURSor|GRID|MARKer|ZBOX}

Function Sets or queries the intensity of a display item.

Syntax
```
:DISPlay:INTENsity:{CURSor|GRID|MARK
er|ZBOX} {<NRf>}
:DISPlay:INTENsity:{CURSor|GRID|MARK
er|ZBOX}?
```
<NRf> = 0 to 31

Example The following example is for the cursor.
```
:DISPLAY:INTENSITY:CURSOR 10
:DISPLAY:INTENSITY:CURSOR?
-> :DISPLAY:INTENSITY:CURSOR 10
```

:DISPlay:INTERpolate

Function Sets or queries the interpolation method.

Syntax
```
:DISPlay:INTERpolate {OFF|LINE|
PULSe|SINE}
:DISPlay:INTERpolate?
```

Example
```
:DISPLAY:INTERPOLATE OFF
:DISPLAY:INTERPOLATE?
-> :DISPLAY:INTERPOLATE OFF
```

:DISPlay:MAPPing

Function Sets or queries the split screen waveform mapping mode.

Syntax
```
:DISPlay:MAPPing {AUTO|MANual}
:DISPlay:MAPPing?
```

Example
```
:DISPLAY:MAPPING AUTO
:DISPLAY:MAPPING?
-> :DISPLAY:MAPPING AUTO
```

:DISPlay:SMAPping?

Function Queries all waveform mappings to the split screen.

Syntax
```
:DISPlay:SMAPping?
```

:DISPlay:SMAPping:{CHANnel<x>|MATH<x>} (Set Mapping)

Function Sets or queries the specified waveform mapping to the split screen.

Syntax
```
:DISPlay:SMAPping:CHANnel<x> {<NRf>}
:DISPlay:SMAPping:CHANnel<x>?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (2 on 2-channel models)
<NRf> = 1 to 8

Example
```
:DISPLAY:SMAPPING:CHANNEL1 1
:DISPLAY:SMAPPING:CHANNEL1?
-> :DISPLAY:SMAPPING:CHANNEL1 1
```

:DISPlay:SVALue (Scale VALUE)

Function Sets or queries the on/off status of the scale value display.

Syntax
```
:DISPlay:SVALue {<Boolean>}
:DISPlay:SVALue?
```

Example
```
:DISPLAY:SVALUE ON
:DISPLAY:SVALUE?
-> :DISPLAY:SVALUE 1
```

5.11 FFT Group

:FFT<x>?

Function Queries all FFT analysis settings.

Syntax `:FFT<x>?`

<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:AVERage?

Function Queries all settings related to the averaging of FFT analysis.

Syntax `:FFT<x>:AVERage?`

<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:AVERage:EWEight

Function Sets or queries the attenuation constant of exponential averaging of FFT analysis.

Syntax `:FFT<x>:AVERage:EWEight {<NRf>}`

`:FFT<x>:AVERage:EWEight?`

<x> = 1 or 2 (1 only for 2-channel models)

<NRf> = 2 to 1024 (in 2n steps)

Example `:FFT1:AVERAGE:EWEIGHT 2`

`:FFT1:AVERAGE:EWEIGHT?`

`-> :FFT1:AVERAGE:EWEIGHT 2`

:FFT<x>:DATA?

Function Queries all the information related to the FFT waveform data that will be sent.

Syntax `:FFT<x>:DATA?`

<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:DATA:BYTeorder

Function Sets or queries the transmission byte order when the FFT waveform data format is binary.

Syntax `:FFT<x>:DATA:BYTeorder {LSBFirst|MSBFirst}`

`:FFT<x>:DATA:BYTeorder?`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:DATA:BYTEORDER LSBFIRST`

`:FFT1:DATA:BYTEORDER?`

`-> :FFT1:DATA:BYTEORDER LSBFIRST`

:FFT<x>:DATA:END

Function Sets or queries the end point of the FFT waveform data that will be sent.

Syntax `:FFT<x>:DATA:END {<NRf>}`

`:FFT<x>:DATA:END?`

<x> = 1 or 2 (1 only for 2-channel models)

<NRf> = 0 to 1250000

Example `:FFT1:DATA:END 125000`

`:FFT1:DATA:END?`

`-> :FFT1:DATA:END 125000`

:FFT<x>:DATA:FORMat

Function Sets or queries the format of the FFT waveform data that will be sent.

Syntax `:FFT<x>:DATA:FORMat {ASCii|BINary}`

`:FFT<x>:DATA:FORMat?`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:DATA:FORMAT ASCII`

`:FFT1:DATA:FORMAT?`

`-> :FFT:DATA:FORMAT ASCII`

:FFT<x>:DATA:LENGth?

Function Queries the total number of data points of the FFT waveform that will be sent.

Syntax `:FFT<x>:DATA:LENGth?`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:DATA:LENGTH?`

`-> :FFT1:DATA:LENGTH 6251`

:FFT<x>:DATA:SEND?

Function Queries the FFT waveform data.

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:DATA:SEND?`

`-> :FFT1:DATA:SEND #6`

(6-digit number of bytes)(data sequence) or <NRf>,<NRf>,...

Description The output format of ":FFT<x>:DATA:SEND?" varies depending on the ":FFT<x>:DATA:FORMat" setting.

(1) When set to "ASCii," the information is returned in this form: <NRf>,<NRf>,...<NRf>

(2) When set to "BINARY," the information is returned in <block data> format. The data of each point is 4 bytes in length, and <block data> can be converted to computed values through floating-point conversion.

:FFT<x>:DATA:STARt

Function Sets or queries the start point of the FFT waveform data that will be sent.

Syntax `:FFT<x>:DATA:STARt {<NRf>}`

`:FFT<x>:DATA:STARt?`

<x> = 1 or 2 (1 only for 2-channel models)

<NRf> = 0 to 125000

Example `:FFT1:DATA:START 0`

`:FFT1:DATA:START?`

`-> :FFT1:DATA:START 0`

:FFT<x>:DISPlay

Function Sets or queries whether FFT analysis is to be performed.

Syntax
```
:FFT<x>:DISPlay {<Boolean>}
:FFT<x>:DISPlay?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:DISPLAY ON
:FFT1:DISPLAY?
-> :FFT1:DISPLAY 1
```

:FFT<x>:HORizontal?

Function Queries all horizontal axis settings of FFT analysis.

Syntax
```
:FFT<x>:HORizontal?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:HORizontal:CSPan?

Function Queries all settings related to the center point and span of the horizontal axis in FFT analysis.

Syntax
```
:FFT<x>:HORizontal:CSPan?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:HORizontal:CSPan:CENTer

Function Sets or queries the center value of the horizontal axis in FFT analysis.

Syntax
```
:FFT<x>:HORizontal:CSPan:CENTer {<Frequency>}
:FFT<x>:HORizontal:CSPan:CENTer?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Frequency> = 0 to 125 G(Hz)

Example
```
:FFT1:HORIZONTAL:CSPAN:CENTER 1HZ
:FFT1:HORIZONTAL:CSPAN:CENTER?
-> :FFT1:HORIZONTAL:CSPAN:
CENTER 1.000E+00
```

:FFT<x>:HORizontal:CSPan:SPAN

Function Sets or queries the span of the horizontal axis in FFT analysis.

Syntax
```
:FFT<x>:HORizontal:CSPan:SPAN {<Frequency>}
:FFT<x>:HORizontal:CSPan:SPAN?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Frequency> = 0 to 125 G(Hz)

Example
```
:FFT1:HORIZONTAL:CSPAN:SPAN 1HZ
:FFT1:HORIZONTAL:CSPAN:SPAN?
-> :FFT1:HORIZONTAL:CSPAN:
SPAN 1.000E+00
```

:FFT<x>:HORizontal:LRIGht?

Function Queries all settings related to the left and right edges of the horizontal axis in FFT analysis.

Syntax
```
:FFT<x>:HORizontal:LRIGht?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:HORizontal:LRIGht:RANGe

Function Sets or queries the range of the left and right edges of the horizontal axis in FFT analysis.

Syntax
```
:FFT<x>:HORizontal:LRIGht:RANGe {<Frequency>,<Frequency>}
:FFT<x>:HORizontal:LRIGht:RANGe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Frequency> = 0 to 125 G(Hz)

Example
```
:FFT1:HORIZONTAL:LRIGHT:RANGE 1HZ,
2HZ
:FFT1:HORIZONTAL:LRIGHT:RANGE?
-> :FFT1:HORIZONTAL:LRIGHT:
RANGE 2.000E+00,1.000E+00
```

:FFT<x>:HORizontal:MODE

Function Sets or queries the horizontal axis mode of FFT analysis.

Syntax
```
:FFT<x>:HORizontal:MODE {AUTO|CSPan|LRIGht}
:FFT<x>:HORizontal:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:HORIZONTAL:MODE AUTO
:FFT1:HORIZONTAL:MODE?
-> :FFT1:HORIZONTAL:MODE AUTO
```

:FFT<x>:LENGth

Function Sets or queries the number of FFT points of FFT analysis.

Syntax
```
:FFT<x>:LENGth {<NRf>}
:FFT<x>:LENGth?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1250, 2500, 12500, 25000, 125000, 250000, 1250000

Example
```
:FFT1:LENGTH 1250
:FFT1:LENGTH?
-> :FFT1:LENGTH 1250
```

:FFT<x>:MEASure?

Function Queries all automated measurement settings of FFT analysis.

Syntax
```
:FFT<x>:MEASure?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:MARKer?

Function Queries all settings related to the marker cursor measurement of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:MARKer[:BASic]?

Function Queries all settings related to the basic marker cursor items of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer:BASic?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:MARKer[:BASic]:ALL

Function Collectively turns on or off all the basic marker cursor items of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:ALL {<Boolean>}
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:MARKER:BASIC:ALL ON
```

:FFT<x>:MEASure:MARKer[:BASic]:DFRequency?

Function Queries all settings related to the frequency between marker cursors of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:DFRequency?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:MARKer[:BASic]:DFRequency:STATe

Function Sets or queries the on/off state of the frequency between marker cursors of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:DFRequency:STATe {<Boolean>}
:FFT<x>:MEASure:MARKer[:BASic]:DFRequency:STATe?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:MARKER:BASIC:DFREQUENCY:STATE ON
:FFT1:MEASURE:MARKER:BASIC:DFREQUENCY:STATE?
-> :FFT1:MEASURE:MARKER:BASIC:DFREQUENCY:STATE 1
```

:FFT<x>:MEASure:MARKer[:BASic]:DFRequency:VALue?

Function Queries the frequency between marker cursors of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:DFRequency:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:MARKER:BASIC:DFREQUENCY:VALUE?
-> :FFT1:MEASURE:MARKER:BASIC:DFREQUENCY:VALUE 1.000E+00
```

:FFT<x>:MEASure:MARKer[:BASic]:DV?

Function Queries all settings related to the level between marker cursors of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:DV?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:MARKer[:BASic]:DV:STATe

Function Sets or queries the on/off state of the level between marker cursors of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:DV:STATe {<Boolean>}
:FFT<x>:MEASure:MARKer[:BASic]:DV:STATe?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:MARKER:BASIC:DV:STATE ON
:FFT1:MEASURE:MARKER:BASIC:DV:STATE?
-> :FFT1:MEASURE:MARKER:BASIC:DV:STATE 1
```

:FFT<x>:MEASure:MARKer[:BASic]:DV:VALue?

Function Queries the level between marker cursors of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:DV:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:MARKER:BASIC:DV:VALUE?
-> :FFT1:MEASURE:MARKER:BASIC:DV:VALUE 1.000E+00
```

:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>?

Function Queries all settings related to the specified marker cursor's frequency of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:STATe

Function Sets or queries the on/off state of the specified marker cursor's frequency of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:STATe {<Boolean>}
:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:STATe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:MARKER:BASIC:FREQUENCY1:STATE ON
:FFT1:MEASURE:MARKER:BASIC:FREQUENCY1:STATE?
-> :FFT1:MEASURE:MARKER:BASIC:FREQUENCY1:STATE 1
```

:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:VALue?

Function Queries the specified marker cursor's frequency of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:MARKER:BASIC:FREQUENCY1:VALUE?
-> :FFT1:MEASURE:MARKER:BASIC:FREQUENCY1:VALUE 1.000E+00
```

:FFT<x>:MEASure:MARKer[:BASic]:POSition<y>

Function Sets or queries the specified marker cursor's position of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:POSition<y> {<NRf>}
:FFT<x>:MEASure:MARKer[:BASic]:POSition<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = –5 div to 5 div

Example
```
:FFT1:MEASURE:MARKER:BASIC:POSITION1 1
:FFT1:MEASURE:MARKER:BASIC:POSITION1?
-> :FFT1:MEASURE:MARKER:BASIC:POSITION1 1.000E+00
```

:FFT<x>:MEASure:MARKer[:BASic]:V<y>?

Function Queries all settings related to the specified marker cursor's level of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:V<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

:FFT<x>:MEASure:MARKer[:BASic]:V<y>:STATe

Function Sets or queries the on/off state of the specified marker cursor's level of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:STATe {<Boolean>}
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:STATe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:MARKER:BASIC:V1:STATE ON
:FFT1:MEASURE:MARKER:BASIC:V1:STATE?
-> :FFT1:MEASURE:MARKER:BASIC:V1:STATE 1
```

:FFT<x>:MEASure:MARKer[:BASic]:V<y>:VALue?

Function Queries the specified marker cursor's level of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:MARKER:BASIC:V1:VALUE?
-> :FFT1:MEASURE:MARKER:BASIC:V1:VALUE 1.000E+00
```

:FFT<x>:MEASure:MODE

Function Sets or queries the automated measurement mode of FFT analysis.

Syntax
```
:FFT<x>:MEASure:MODE {MARKer|OFF|PEAK}
:FFT<x>:MEASure:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:MODE MARKER
:FFT1:MEASURE:MODE?
-> :FFT1:MEASURE:MODE MARKER
```

:FFT<x>:MEASure:PEAK?

Function Queries all settings related to the peak value measurement of FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:PEAK:DETail?

Function Queries all settings related to the FFT analysis result list.

Syntax
```
:FFT<x>:MEASure:PEAK:DETail?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:PEAK:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the FFT analysis result list.

Syntax
```
:FFT<x>:MEASure:PEAK:DETail:LIST:ITEM?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:PEAK:DETAIL:LIST:ITEM?
-> :FFT1:MEASURE:PEAK:DETAIL:LIST:ITEM? "No,Frequency,Peak"
```

:FFT<x>:MEASure:PEAK:DETail:LIST:MAXPeak:EXECute

Function Moves to the maximum peak value in the FFT analysis results.

Syntax
```
:FFT<x>:MEASure:PEAK:DETail:LIST:MAXPeak:EXECute
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:PEAK:DETAIL:LIST:MAXPeak:EXECUTE
```

:FFT<x>:MEASure:PEAK:DETail:LIST:NUMBer

Function Sets or queries the analysis number in the FFT analysis result list.

Syntax
```
:FFT<x>:MEASure:PEAK:DETail:LIST:NUMBer {<NRf>}
:FFT<x>:MEASure:PEAK:DETail:LIST:NUMBer?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 10

Example
```
:FFT1:MEASURE:PEAK:DETAIL:LIST:NUMBER 1
:FFT1:MEASURE:PEAK:DETAIL:LIST:NUMBER?
-> :FFT1:MEASURE:PEAK:DETAIL:LIST:NUMBER 1
```

:FFT<x>:MEASure:PEAK:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the FFT analysis result list.

Syntax
```
:FFT<x>:MEASure:PEAK:DETail:LIST:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:PEAK:DETAIL:LIST:VALUE?
```

:FFT<x>:MEASure:PEAK:EXCursion

Function Sets or queries the difference between peak and valley of the peak value of FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK:EXCursion {<NRf>}
:FFT<x>:MEASure:PEAK:EXCursion?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:FFT1:MEASURE:PEAK:EXCURSION 1
:FFT1:MEASURE:PEAK:EXCURSION?
-> :FFT1:MEASURE:PEAK:EXCURSION 1.000E+00
```

:FFT<x>:MEASure:PEAK:FVALue?

Function Queries the peak frequency of FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK:FVALue?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:PEAK:FVALUE?
-> :FFT1:MEASURE:PEAK:FVALUE 1.000E+00
```

:FFT<x>:MEASure:PEAK:THReshold

Function Sets or queries the threshold level of the peak value of FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK:THReshold {<NRf>}
:FFT<x>:MEASure:PEAK:THReshold?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:FFT1:MEASURE:PEAK:THRESHOLD 1
:FFT1:MEASURE:PEAK:THRESHOLD?
-> :FFT1:MEASURE:PEAK:THRESHOLD 1.000E+00
```

:FFT<x>:MEASure:PEAK:VVALue?

Function Queries the peak value of FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK:VVALue?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MEASURE:PEAK:VVALUE?
-> :FFT1:MEASURE:PEAK:VVALUE 1.000E+00
```

:FFT<x>:MODE

Function Sets or queries the waveform display mode of FFT analysis.

Syntax
```
:FFT<x>:MODE {AVERage|MAXHold|NORMal}
:FFT<x>:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:MODE AVERAGE
:FFT1:MODE?
-> :FFT1:MODE AVERAGE
```

:FFT<x>:RANGe

Function Sets or queries the measurement source window of FFT analysis.

Syntax
```
:FFT<x>:RANGe {MAIN|Z1|Z2}
:FFT<x>:RANGe?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:RANGE MAIN
:FFT1:RANGE?
-> :FFT1:RANGE MAIN
```

:FFT<x>:RPOSition

Function Sets or queries the center zoom point on the vertical axis in FFT analysis.

Syntax
```
:FFT<x>:RPOSition {<NRf>}
:FFT<x>:RPOSition?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = –4 div to 4 div

Example
```
:FFT1:RPOSITION 1
:FFT1:RPOSITION?
-> :FFT1:RPOSITION 1.000E+00
```

:FFT<x>:TRACe

Function Sets or queries the source waveform of FFT analysis.

Syntax
```
:FFT<x>:TRACe {<NRf>|MATH<y>}
:FFT<x>:TRACe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:FFT1:TRACE 1
:FFT1:TRACE?
-> :FFT1:TRACE 1
```

:FFT<x>:TYPE

Function Sets or queries the spectrum of FFT analysis.

Syntax
```
:FFT<x>:TYPE {CH_MAG|CS_IMAG|CS_
LOGMAG|CS_MAG|CS_PHASE|CS_REAL|LS_
IMAG|LS_LOGMAG|LS_MAG|LS_PHASE|LS_
REAL|PS_LOGMAG|PS_MAG|PSD_
LOGMAG|PSD_MAG|RS_LOGMAG|RS_MAG|TF_
IMAG|TF_LOGMAG|TF_MAG|TF_PHASE|TF_
REAL[,<NRf>|MATH<y>]}
:FFT<x>:TYPE?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:FFT1:TYPE CH_MAG,1
:FFT1:TYPE?
-> :FFT1:TYPE CH_MAG,1
```

Description
- An error will occur if the user-defined computation (option) is not available.
- When the spectrum is set to CH_MAG|CS_IMAG|CS_LOGMAG|CS_MAG|CS_PHASE|CS_REAL|TF_IMAG|TF_LOGMAG|TF_MAG|TF_PHASE|TF_REAL, two source waveforms are required. Use :FFT<x>:TRACe to set the first waveform, and use <NRf> or MATH<y> of this command to set the second waveform. If the spectrum is set to any other value, the source waveform specified with the :FFT<x>:TRACe command is valid.

:FFT<x>:UNIT?

Function Queries all FFT computation unit settings.

Syntax
```
:FFT<x>:UNIT?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:UNIT[:DEFine]

Function Sets or queries the FFT computation unit.

Syntax
```
:FFT<x>:UNIT[:DEFine] {<String>}
:FFT<x>:UNIT[:DEFine]?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:UNIT:DEFINE "EU"
:FFT1:UNIT:DEFINE?
-> :FFT1:UNIT:DEFINE "EU"
```

Description The unit is applied to scaled values. This command has no effect on the computation result.

:FFT<x>:UNIT:MODE

Function Sets or queries whether an FFT computation unit will be attached automatically or manually.

Syntax
```
:FFT<x>:UNIT:MODE {AUTO|USERdefine}
:FFT<x>:UNIT:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:UNIT:MODE AUTO
:FFT1:UNIT:MODE?
-> :FFT1:UNIT:MODE AUTO
```

:FFT<x>:VERTical?

Function Queries all vertical axis settings of FFT analysis.

Syntax
```
:FFT<x>:VERTical?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:VERTical:LEVel

Function Sets or queries the vertical axis display position of FFT analysis.

Syntax
```
:FFT<x>:VERTical:LEVel {<NRf>}
:FFT<x>:VERTical:LEVel?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:FFT1:VERTICAL:LEVEL 1
:FFT1:VERTICAL:LEVEL?
-> :FFT1:VERTICAL:LEVEL 1.000E+00
```

:FFT<x>:VERTical:MODE

Function Sets or queries the vertical axis mode of FFT analysis.

Syntax
```
:FFT<x>:VERTical:MODE {AUTO|MANual}
:FFT<x>:VERTical:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:VERTICAL:MODE AUTO
:FFT1:VERTICAL:MODE?
-> :FFT1:VERTICAL:MODE AUTO
```

:FFT<x>:VERTical:SENSitivity

Function Sets or queries the vertical axis sensitivity of FFT analysis.

Syntax
```
:FFT<x>:VERTical:SENSitivity {<NRf>}
:FFT<x>:VERTical:SENSitivity?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 0 to 1.0000E+31

Example
```
:FFT1:VERTICAL:SENSITIVITY 1
:FFT1:VERTICAL:SENSITIVITY?
-> :FFT1:VERTICAL:
SENSITIVITY 1.000E+00
```

:FFT<x>:VTDisplay

Function Sets or queries the on/off state of the VT waveform display of FFT analysis.

Syntax
```
:FFT<x>:VTDisplay {<Boolean>}
:FFT<x>:VTDisplay?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:VTDISPLAY ON
:FFT1:VTDISPLAY?
-> :FFT1:VTDISPLAY 1
```

:FFT<x>:WINDow

Function Sets or queries the window function of FFT analysis.

Syntax
```
:FFT<x>:WINDow {FLATtop|HANNing|
RECTangle}
:FFT<x>:WINDow?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:WINDOW FLATTOP
:FFT1:WINDOW?
-> :FFT1:WINDOW FLATTOP
```

5.12 FILE Group

:FILE?

Function Queries all data storage settings.
Syntax :FILE?

:FILE:COPY:ABORt

Function Aborts file copying.
Syntax :FILE:COPY:ABORt
Example :FILE:COPY:ABORT

:FILE:COPY:CDIRectory

Function Changes the file copy destination directory.
Syntax :FILE:COPY:CDIRectory {<String>}
<String> = See the Features Guide for this information.
Example :FILE:COPY:CDIRECTORY "UTIL"
(Relative path designation)

:FILE:COPY:DRIVe

Function Sets the storage device to copy files to.
Syntax :FILE:COPY:DRIVe {FLAShmem|USB,<NRf>}
<NRf> = 0 to 3
Example :FILE:COPY:DRIVE FLASHMEM
Description The <NRf> for USB can be omitted if the drive is not partitioned or divided by LUN.

:FILE:COPY[:EXECute]

Function Executes file copying. This is an overlap command.
Syntax :FILE:COPY[:EXECute] {<String>}
<String> = See the Features Guide for this information.
Example :FILE:COPY:EXECUTE "DATA.PNG"

:FILE:COPY:PATH?

Function Queries the file copy destination directory.
Syntax :FILE:COPY:PATH?
Example :FILE:COPY:PATH?
-> :FILE:COPY:PATH "PATH = FLASHMEM/UTIL"

:FILE:DELete:{AHIStogram|ASCii|BINary|BMP|FFT|HLISt|JPEG|MEASure|PNG|SBUS|SETup|SNAP|ZPOLygon|ZWAVe}[:EXECute]

Function Deletes various types of data files. This is an overlap command.
Syntax :FILE:DELete:{AHIStogram|ASCii|BINary|BMP|FFT|HLISt|JPEG|MEASure|PNG|SBUS|SETup|SNAP|ZPOLygon|ZWAVe}[:EXECute] {<Filename>}
Example The following example is for waveform data.
:FILE:DELETE:AHISTOGRAM:EXECUTE "DATA"
Description Use the :FILEDIRectory:DRIVe to select the target storage device.

:FILE[:DIRectory]:CDIRectory

Function Changes the current directory.
Syntax :FILE[:DIRectory]:CDIRectory {<String>}
<String> = See the Features Guide for this information.
Example :FILE:DIRECTORY:CDIRECTORY "UTIL"

:FILE[:DIRectory]:DRIVe

Function Sets the current storage device.
Syntax :FILE[:DIRectory]:DRIVe {FLAShmem|NETWork|USB,<NRf>}
<NRf> = 0 to 3
Example :FILE:DIRECTORY:DRIVE FLASHMEM
Description The <NRf> for USB can be omitted if the drive is not partitioned or divided by LUN.

:FILE[:DIRectory]:FREE?

Function Queries the free space on the current storage device that is being operated on in bytes.
Syntax :FILE[:DIRectory]:FREE?
Example :FILE:DIRECTORY:FREE?
-> :FILE:DIRECTORY:FREE 65536

:FILE[:DIRectory]:MDIRectory

Function Creates a directory in the current location. This is an overlap command.
Syntax :FILE[:DIRectory]:MDIRectory {<String>}
<String> = See the Features Guide for this information.
Example :FILE:DIRECTORY:MDIRECTORY "DIR2"

:FILE[:DIRectory]:PATH?
Function Queries the current directory.
Syntax :FILE[:DIRectory]:PATH?
Example :FILE:DIRECTORY:PATH?
-> :FILE:DIRECTORY:
PATH "PATH = FLASHMEM/UTIL"

:FILE:LOAD:BINary:ABORt
Function Aborts the loading of waveform data.
Syntax :FILE:LOAD:BINary:ABORt
Example :FILE:LOAD:BINARY:ABORT

:FILE:LOAD:BINary[:EXECute]
Function Executes the loading of waveform data. This is an overlap command.
Syntax :FILE:LOAD:BINary[:
EXECute] {<Filename>{,ACQMemory|
REFerence<x>}}
<x> = 1 or 2
Example :FILE:LOAD:BINARY:EXECUTE "DATA"

:FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}:ABORt
Function Aborts the loading of various types of data.
Syntax :FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}:ABORt
<x> = 1 to 4
Example The following example is for setup data.
:FILE:LOAD:SETUP:ABORT

:FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}[:EXECute]
Function Executes the loading of various types of data. This is an overlap command.
Syntax :FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}[:EXECute] {<Filename>}
<x> = 1 to 4
Example The following example is for setup data.
:FILE:LOAD:SETUP:EXECUTE "DATA"

:FILE:MOVE:ABORt
Function Aborts file moving.
Syntax :FILE:MOVE:ABORt
Example :FILE:MOVE:ABORT

:FILE:MOVE:CDIRectory
Function Changes the file move destination directory.
Syntax :FILE:MOVE:CDIRectory {<String>}
<String> = See the Features Guide for this information.
Example :FILE:MOVE:CDIRECTORY "UTIL"

:FILE:MOVE:DRIVe
Function Sets the storage device to move files to.
Syntax :FILE:MOVE:DRIVe {FLAShmem|
USB,<NRf>}
<NRf> = 0 to 3
Example :FILE:MOVE:DRIVE FLASHMEM
Description The <NRf> for USB can be omitted if the drive is not partitioned or divided by LUN.

:FILE:MOVE[:EXECute]
Function Executes file moving. This is an overlap command.
Syntax :FILE:MOVE[:EXECute] {<String>}
<String> = See the Features Guide for this information.
Example :FILE:MOVE:EXECUTE "DATA.PNG"

:FILE:MOVE:PATH?
Function Queries the file move destination directory.
Syntax :FILE:MOVE:PATH?
Example :FILE:MOVE:PATH?
-> :FILE:MOVE:PATH "PATH = FLASHMEM/UTIL"

:FILE:PROTect[:EXECute]
Function Turns on or off the file protection.
Syntax :FILE:PROTect[:EXECute] {<String>,<Boolean>}
<String> = See the Features Guide for this information.
Example :FILE:PROTECT:EXECUTE "DATA.PNG",ON

:FILE:REName[:EXECute]
Function Renames the file.
Syntax :FILE:REName[:EXECute] {<String>,<String>}
<String> = See the Features Guide for this information.
Example :FILE:RENAME:EXECUTE "DATA.
PNG","000.PNG"

:FILE:SAVE?
Function Queries all file save settings.
Syntax :FILE:SAVE?

:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|SBUS|ZWAVe}?
Function Queries all the settings related to the saving of a specific type of file.
Syntax :FILE:SAVE:{AHIStogram|ASCii|BINary|
FFT|HLISt|HARMonics|SBUS|ZWAVe}?

:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}:ABORt

Function Aborts the saving of a specific type of file.

Syntax
```
:FILE:SAVE:{AHIStogram|ASCii|BINary|
FFT|HLISt|HARMonics|MEASure|SBUS|SET
up|SNAP|ZWAVe}:ABORt
```

Example The following example is for waveform data.
```
:FILE:SAVE:BINARY:ABORT
```

:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}[:EXECute]

Function Executes the saving of a specific type of file. This is an overlap command.

Syntax
```
:FILE:SAVE:{AHIStogram|ASCii|BINary|
FFT|HARMonics|MEASure|SBUS|SETup|SNA
P|ZWAVe}[:EXECute] {<Filename>}
```

Example The following example is for waveform data.
```
:FILE:SAVE:BINARY:EXECUTE "DATA"
```

:FILE:SAVE:{AHIStogram|FFT|HARMonics|SBUS|ZWAVe}:SELect

Function Sets or queries the area for saving a specific type of file.

Syntax
```
:FILE:SAVE:AHIStogram:SELect {<NRf>}
:FILE:SAVE:AHIStogram:SELect?
```
<NRf> = 1 or 2 (1 to 4 for SBUS and ZWAVe)

Example
```
The following example is for FFT.
:FILE:SAVE:FFT:SELECT 1
:FILE:SAVE:FFT:SELECT?
-> :FILE:SAVE:FFT:SELECT 1
```

:FILE:SAVE:ANAMing

Function Sets or queries the on/off state of the auto generation of the name of the file to be saved.

Syntax
```
:FILE:SAVE:ANAMing {DATE|DATE2|
NUMBering|OFF|ON}
:FILE:SAVE:ANAMing?
```

Example
```
:FILE:SAVE:ANAMING DATE
:FILE:SAVE:ANAMING?
-> :FILE:SAVE:ANAMING DATE
```

:FILE:SAVE:{ASCii|BINary}:COMPression

Function Sets or queries the save compression method for a specific type of file.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:COMPressi
on {DECimation|OFF|PTOPeak}
:FILE:SAVE:{ASCii|BINary}:COMPressi
on?
```

Example The following example is for waveform data.
```
:FILE:SAVE:BINARY:
COMPRESSION DECIMATION
:FILE:SAVE:BINARY:COMPRESSION?
-> :FILE:SAVE:BINARY:
COMPRESSION DECIMATION
```

:FILE:SAVE:{ASCii|BINary}:HISTory

Function Sets or queries the save method for the specified history waveform.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:
HISTory {ALL|ONE}
:FILE:SAVE:{ASCii|BINary}:HISTory?
```

Example The following example is for waveform data.
```
:FILE:SAVE:BINARY:HISTORY ALL
:FILE:SAVE:BINARY:HISTORY?
-> :FILE:SAVE:BINARY:HISTORY ALL
```

:FILE:SAVE:{ASCii|BINary}:LENGth

Function Sets or queries the number of data points to save when compressing or sampling (decimating) various types of data.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:LENG
th {<NRf>}
:FILE:SAVE:{ASCii|BINary}:LENGth?
```
<NRf> = See the Features Guide for this information.

Example The following example is for waveform data.
```
:FILE:SAVE:BINARY:LENGTH 1250
:FILE:SAVE:BINARY:LENGTH?
-> :FILE:SAVE:BINARY:LENGTH 1250
```

:FILE:SAVE:{ASCii|BINary}:RANGe

Function Sets or queries the window for saving a specific type of data.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:
RANGe {MAIN|Z1|Z2}
:FILE:SAVE:{ASCii|BINary}:RANGe?
```

Example The following example is for waveform data.
```
:FILE:SAVE:BINARY:RANGE MAIN
:FILE:SAVE:BINARY:RANGE?
-> :FILE:SAVE:BINARY:RANGE MAIN
```

:FILE:SAVE:{ASCii|BINary}:TRACe

Function Sets or queries the waveform for saving a specific type of data.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:
TRACe {<NRf>|ALL|MATH<x>}
:FILE:SAVE:{ASCii|BINary}:TRACe?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:FILE:SAVE:BINARY:TRACE 1
:FILE:SAVE:BINARY:TRACE?
-> :FILE:SAVE:ASCII:TRACE 1
```

:FILE:SAVE:ASCii:TINFormation (Time Information)

Function Sets or queries whether waveform data is saved with time information (ON) or without it (OFF).

Syntax
```
:FILE:SAVE:ASCii:
TINFormation {<Boolean>}
:FILE:SAVE:ASCii:TINFormation?
```

Example
```
:FILE:SAVE:ASCII:TINFORMATION ON
:FILE:SAVE:ASCII:TINFORMATION?
-> :FILE:SAVE:ASCII:TINFORMATION 1
```

:FILE:SAVE:COMMent

Function Sets or queries the comment for the data to be saved.

Syntax
```
:FILE:SAVE:COMMent {<String>}
:FILE:SAVE:COMMent?
```
<String> = Up to 128 characters

Example
```
:FILE:SAVE:COMMENT "THIS IS TEST."
:FILE:SAVE:COMMENT?
-> :FILE:SAVE:COMMENT "THIS IS
TEST."
```

Description You can only use the characters and symbols on the keyboard that appears on the instrument's screen.

:FILE:SAVE:FFT:FINFormation (FrequencyInformation)

Function Sets or queries whether FFT data is saved with frequency information (ON) or without it (OFF).

Syntax
```
:FILE:SAVE:FFT:
FINFormation {<Boolean>}
:FILE:SAVE:FFT:FINFormation?
```

Example
```
:FILE:SAVE:FFT:FINFORMATION ON
:FILE:SAVE:FFT:FINFORMATION?
-> :FILE:SAVE:FFT:FINFORMATION 1
```

:FILE:SAVE:NAME

Function Sets or queries the file name for the data to be saved.

Syntax
```
:FILE:SAVE:NAME {<Filename>}
:FILE:SAVE:NAME?
```

Example
```
:FILE:SAVE:NAME "CASE1"
:FILE:SAVE:NAME?
-> :FILE:SAVE:NAME "CASE1"
```

:FILE:SAVE:SBUS:HISTory

Function Sets or queries the history for which serial bus data will be saved.

Syntax
```
:FILE:SAVE:SBUS:HISTory {ALL|ONE}
:FILE:SAVE:SBUS:HISTory?
```

Example
```
:FILE:SAVE:SBUS:HISTORY ALL
:FILE:SAVE:SBUS:HISTORY?
-> :FILE:SAVE:SBUS:HISTORY ALL
```

5.13 GONogo Group

:GONogo?

Function Queries all GO/NO-GO determination settings.

Syntax `:GONogo?`

:GONogo:ABORt

Function Aborts GO/NO-GO determination.

Syntax `:GONogo:ABORt`

Example `:GONOGO:ABORT`

:GONogo:ACTion?

Function Queries all settings for the action that is performed when the judgment is NO-GO and the reference for the judgment.

Syntax `:GONogo:ACTion?`

:GONogo:ACTion:BUZZer

Function Sets or queries whether a warning sound is to be generated when the judgment is NO-GO.

Syntax `:GONogo:ACTion:BUZZer {<Boolean>}`
`:GONogo:ACTion:BUZZer?`

Example `:GONOGO:ACTION:BUZZER ON`
`:GONOGO:ACTION:BUZZER?`
`-> :GONOGO:ACTION:BUZZER 1`

:GONogo:ACTion:HCOPy

Function Sets or queries whether a hard copy is output to the built-in printer (option) or an external printer when the judgment is NO-GO.

Syntax `:GONogo:ACTion:HCOPy {<Boolean>}`
`:GONogo:ACTion:HCOPy?`

Example `:GONOGO:ACTION:HCOPY ON`
`:GONOGO:ACTION:HCOPY?`
`-> :GONOGO:ACTION:HCOPY 1`

:GONogo:ACTion:MAIL?

Function Queries all settings related to the mail notification that is made when the judgment is NO-GO.

Syntax `:GONogo:ACTion:MAIL?`

:GONogo:ACTion:MAIL:COUNt

Function Sets or queries the upper limit to the number of mail notifications made when the judgment is NO-GO.

Syntax `:GONogo:ACTion:MAIL:COUNt {<NRf>}`
`:GONogo:ACTion:MAIL:COUNt?`
<NRf> = 1 to 1000

Example `:GONOGO:ACTION:MAIL:COUNT 100`
`:GONOGO:ACTION:MAIL:COUNT?`
`-> :GONOGO:ACTION:MAIL:COUNT 100`

:GONogo:ACTion:MAIL:MODE

Function Sets or queries whether a mail notification is to be made when the judgment is NO-GO.

Syntax `:GONogo:ACTion:MAIL:MODE {<Boolean>}`
`:GONogo:ACTion:MAIL:MODE?`

Example `:GONOGO:ACTION:MAIL:MODE ON`
`:GONOGO:ACTION:MAIL:MODE?`
`-> :GONOGO:ACTION:MAIL:MODE 1`

:GONogo:ACTion:SAVE

Function Sets or queries whether the event is to be saved to the storage device when the judgment is NO-GO.

Syntax `:GONogo:ACTion:SAVE {<Boolean>}`
`:GONogo:ACTion:SAVE?`

Example `:GONOGO:ACTION:SAVE ON`
`:GONOGO:ACTION:SAVE?`
`-> :GONOGO:ACTION:SAVE 1`

Description Use :FILE:DIRECTORY:DRIVE to set or query the type of storage device.

:GONogo:COUNt?

Function Queries the number of GO/NO-GO determinations that were performed.

Syntax `:GONogo:COUNt?`

Example `:GONOGO:COUNT?`
`-> :GONOGO:COUNT 100`

:GONogo:EXECute

Function Executes GO/NO-GO determination. This is an overlap command.

Syntax `:GONogo:EXECute`

Example `:GONOGO:EXECUTE`

:GONogo:LOGic

Function Sets or queries the GO/NO-GO determination type.

Syntax `:GONogo:LOGic {AND|OFF|OR}`
`:GONogo:LOGic?`

Example `:GONOGO:LOGIC AND`
`:GONOGO:LOGIC?`
`-> :GONOGO:LOGIC AND`

:GONogo:NGCount?

Function Queries the GO/NO-GO determination NO-GO count.

Syntax `:GONogo:NGCount?`

Example `:GONOGO:NGCOUNT?`
`-> :GONOGO:NGCOUNT 5`

:GONogo:NGStopcount
Function Sets or queries the number of NO-GO results.
Syntax
```
:GONogo:NGStopcount {<NRf>|INFinity}
:GONogo:NGStopcount?
```
<NRf> = 1 to 1000
Example
```
:GONOGO:NGSTOPCOUNT 100
:GONOGO:NGSTOPCOUNT?
-> :GONOGO:NGSTOPCOUNT 100
```

:GONogo:STOPcount
Function Sets or queries the number of determination results acquired.
Syntax
```
:GONogo:STOPcount {<NRf>|INFinity}
:GONogo:STOPcount?
```
<NRf> = 1 to 1000000
Example
```
:GONOGO:STOPCOUNT 1000
:GONOGO:STOPCOUNT?
-> :GONOGO:STOPCOUNT 1000
```

:GONogo:WAIT?
Function Waits for the completion of GO/NO-GO determination with a timeout.
Syntax
```
:GONogo:WAIT? {<NRf>}
```
<NRf> = 0 to 864000 (100 ms steps)
Example
```
(Set the timeout period to 5 s.)
:GONOGO:WAIT? 50
-> :GONOGO:WAIT 0
```
Description The instrument returns 0 if the operation finishes before the timer expires and returns 1 if the timer expires.

:GONogo[:ZPARameter]?
Function Queries all zone or parameter determination settings.
Syntax
```
:GONogo[:ZPARameter]?
```

:GONogo[:ZPARameter]:NUMBer<x>?
Function Queries all settings related to the zone or parameter determination conditions.
Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>?
```
<x> = 1 to 4

:GONogo[:ZPARameter]:NUMBer<x>:CAUSe?
Function Queries whether the specified waveform parameter for zone or parameter determination is the cause of a NO-GO judgment.
Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:CAUSe?
```
<x> = 1 to 4
Example
```
:GONOGO:ZPARAMETER:NUMBER1:CAUSE?
-> :GONOGO:ZPARAMETER:NUMBER1:CAUSE 1
```
Description When the parameter is the cause of a NO-GO result, the instrument returns 1. Otherwise, the instrument returns 0.

:GONogo[:ZPARameter]:NUMBer<x>:CONDition
Function Sets or queries the specified waveform parameter's reference condition for zone or parameter determination.
Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:CONDition {IN|OUT}
:GONogo[:ZPARameter]:NUMBer<x>:CONDition?
```
<x> = 1 to 4
Example
```
:GONOGO:ZPARAMETER:NUMBER1:CONDITION IN
:GONOGO:ZPARAMETER:NUMBER1:CONDITION?
-> :GONOGO:ZPARAMETER:NUMBER1:CONDITION IN
```

:GONogo[:ZPARameter]:NUMBer<x>:MODE
Function Sets or queries the mode of the specified condition.
Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:MODE {OFF|PARameter|POLYgon|RECTangle|WAVE}
:GONogo[:ZPARameter]:NUMBer<x>:MODE?
```
<x> = 1 to 4
Example
```
:GONOGO:ZPARAMETER:NUMBER1:MODE PARAMETER
:GONOGO:ZPARAMETER:NUMBER1:MODE?
-> :GONOGO:ZPARAMETER:NUMBER1:MODE PARAMETER
```

:GONogo[:ZPARameter]:NUMBer<x>:PARameter?
Function Queries all the parameter settings of the specified condition.
Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:PARameter?
```
<x> = 1 to 4

:GONogo[:ZPARameter]:NUMBer<x>:PARameter:ITEM

Function Sets or queries the specified waveform parameter's item for parameter determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:ITEM {<Parameter>}
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:ITEM?
```
<x> = 1 to 4
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

Example
```
:GONOGO:ZPARAMETER:NUMBER1:PARAMETER:ITEM AVERAGE
:GONOGO:ZPARAMETER:NUMBER1:PARAMETER:ITEM?
-> :GONOGO:ZPARAMETER:NUMBER1:PARAMETER:ITEM AVERAGE
```

:GONogo[:ZPARameter]:NUMBer<x>:PARameter:LIMit

Function Sets or queries the specified waveform parameter's upper and lower limits for parameter determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:LIMit {<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:LIMit?
```
<x> = 1 to 4
<NRf> = See the Features Guide for this information.

Example
```
:GONOGO:ZPARAMETER:NUMBER1:PARAMETER:LIMIT 0,1
:GONOGO:ZPARAMETER:NUMBER1:PARAMETER:LIMIT?
-> :GONOGO:ZPARAMETER:NUMBER1:PARAMETER:LIMIT 1.000E+00,0.000E+00
```

:GONogo[:ZPARameter]:NUMBer<x>:PARameter:TRACe

Function Sets or queries the specified waveform parameter's source waveform for parameter determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:TRACe {<NRf>|BIT<y>|FFT<y>|MATH<y>|XY<y>}
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of BIT<y> = 1 to 8
<y> of FFT<y> = 1 or 2 (1 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of XY<y> = 1 or 2 (1 on 2-channel models)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:PARAMETER:TRACE 1
:GONOGO:ZPARAMETER:NUMBER1:PARAMETER:TRACE?
-> :GONOGO:ZPARAMETER:NUMBER1:PARAMETER:TRACE 1
```

:GONogo[:ZPARameter]:NUMBer<x>:PARameter:VALue?

Function Queries the measured value of the specified waveform parameter for parameter determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:VALue?
```
<x> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:PARAMETER:VALUE?
-> :GONOGO:ZPARAMETER:NUMBER1:PARAMETER:VALUE 1.98E-03
```

Description If the mode is off or the value cannot be measured, the instrument returns "NAN" (not a number).

:GONogo[:ZPARameter]:NUMBer<x>:POLYgon?

Function Queries all polygonal zone determination settings.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon?
```
<x> = 1 to 4

:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:HPOSition

Function Sets or queries the horizontal position used in polygonal zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:HPOSition {<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:HPOSition?
```
<x> = 1 to 4
<NRf> = –5 div to 5 div

Example
```
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:HPOSITION 1
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:HPOSITION?
-> :GONOGO:ZPARAMETER:NUMBER1:POLYGON:HPOSITION 1.000E+00
```

Description When TRACe is XY<x>, <NRf> = –4 div to 4 div.

:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:RANGe

Function Sets or queries the source window used in polygonal zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:RANGe {MAIN|Z1|Z2}
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:RANGe?
```
<x> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:RANGE MAIN
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:RANGE?
-> :GONOGO:ZPARAMETER:NUMBER1:POLYGON:RANGE MAIN
```

:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:TRACe

Function Sets or queries the source waveform used in polygonal zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:TRACe {<NRf>|MATH<y>|XY<y>}
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of XY<y> = 1 or 2 (1 on 2-channel models)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:TRACE 1
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:TRACE?
-> :GONOGO:ZPARAMETER:NUMBER1:POLYGON:TRACE 1
```

:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:VPOSition

Function Sets or queries the vertical position used in polygonal zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:VPOSition {<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:VPOSition?
```
<x> = 1 to 4
<NRf> = –4 div to 4 div

Example
```
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:VPOSITION 1
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:VPOSITION?
-> :GONOGO:ZPARAMETER:NUMBER1:POLYGON:VPOSITION 1.000E+00
```

:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:ZNUMber

Function Sets or queries the zone number used in polygonal zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:ZNUMber {<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:ZNUMber?
```
<x> = 1 to 4
<NRf> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:ZNUMBER 1
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:ZNUMBER?
-> :GONOGO:ZPARAMETER:NUMBER1:POLYGON:ZNUMBER 1
```

:GONogo[:ZPARameter]:NUMBer<x>:RECTangle?

Function Queries all waveform rectangular zone determination settings.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle?
```
<x> = 1 to 4

:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:HORizontal

Function Sets or queries the horizontal position of the rectangle used in rectangular zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:HORizontal {<NRf>,<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:HORizontal?
```
<x> = 1 to 4
<NRf> = –5 div to 5 div

Example
```
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:HORIZONTAL 1,2
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:HORIZONTAL?
-> :GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:HORIZONTAL 2.000E+00,1.000E+00
```

Description When TRACe is XY<x>, <NRf> = –4 div to 4 div.

:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:RANGe

Function Sets or queries the source window of the rectangle used in rectangular zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:RANGe {MAIN|Z1|Z2}
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:RANGe?
```
<x> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:RANGE MAIN
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:RANGE?
-> :GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:RANGE MAIN
```

:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:TRACe

Function Sets or queries the source waveform of the rectangle used in rectangular zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:TRACe {<NRf>|MATH<y>|XY<y>}
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of XY<y> = 1 or 2 (1 on 2-channel models)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:TRACE 1
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:TRACE?
-> :GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:TRACE 1
```

:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:VERTical

Function Sets or queries the vertical position of the rectangle used in rectangular zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:VERTical {<NRf>,<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:VERTical?
```
<x> = 1 to 4
<NRf> = –4 div to 4 div

Example
```
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:VERTICAL 1,2
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:VERTICAL?
-> :GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:VERTICAL 2.000E+00,1.000E+00
```

:GONogo[:ZPARameter]:NUMBer<x>:WAVE?

Function Queries all waveform zone determination settings.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE?
```
<x> = 1 to 4

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:EXIT

Function Exits from the waveform zone edit menu.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:EXIT {QUIT|STORe}
```
<x> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:EDIT:EXIT STORE
```

Description When STORe is specified, the edited contents are saved, and the edit menu is closed.
When QUIT is specified, the edit menu is closed without saving the edited contents.

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:NEW

Function Sets the base waveform in waveform zone editing.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:NEW {<NRF>|MATH<y>}
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:EDIT:NEW 1
```

Description When you finish editing, you need to use ":GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:EXIT" to exit from the edit menu.

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:PART

Function Executes partial editing in waveform zone editing.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:PART {<NRf>,<NRf>,<NRf>,<NRf>}
```
<x> = 1 to 4
<NRf> = –5.00 to 5.00 (div, T_Range1/T_RAnge2), –8.00 to 8.00 (div (difference), up and down)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:EDIT:PART -2.00,2.00,1.00,1.00
```

Description When you finish editing, you need to use ":GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:EXIT" to exit from the edit menu.

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:WHOLe

Function Sets the entire waveform zone in waveform zone editing.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:WHOLe {<NRf>,<NRf>,<NRf>,<NRf>}
```
<x> = 1 to 4
<NRf> = –5.00 to 5.00 (div, left and right), 0 to 8.00 (div, up and down)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:EDIT:WHOLE 0.50,0.50,1.00,1.00
```

Description When you finish editing, you need to use ":GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:EXIT" to exit from the edit menu.

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:RANGe

Function Sets or queries the source window used in waveform zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:RANGe {MAIN|Z1|Z2}
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:RANGe?
```
<x> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:RANGE MAIN
:GONOGO:ZPARAMETER:NUMBER1:WAVE:RANGE?
-> :GONOGO:ZPARAMETER:NUMBER1:WAVE:RANGE MAIN
```

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRACe

Function Sets or queries the source waveform used in waveform zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRACe {<NRf>|MATH<y>}
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:TRACE 1
:GONOGO:ZPARAMETER:NUMBER1:WAVE:TRACE?
-> :GONOGO:ZPARAMETER:NUMBER1:WAVE:TRACE 1
```

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRANGe

Function Sets or queries the determination period used in waveform zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRANGe {<NRf>,<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRANGe?
```
<NRf> = –5 div to 5 div

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:TRANGE 1,2
:GONOGO:ZPARAMETER:NUMBER1:WAVE:TRANGE?
-> :GONOGO:ZPARAMETER:NUMBER1:WAVE:TRANGE 2.000E+00,1.000E+00
```

:GONogo[:ZPARameter]:NUMBer<x>:WAVE:ZNUMber

Function Sets or queries the zone number used in waveform zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:ZNUMber {<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:ZNUMber?
```
<x> = 1 to 4
<NRf> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:WAVE:ZNUMBER 1
:GONOGO:ZPARAMETER:NUMBER1:WAVE:ZNUMBER?
-> :GONOGO:ZPARAMETER:NUMBER1:WAVE:ZNUMBER 1
```

5.14 HCOPy Group

:HCOPy?

Function Queries all screen capture data output settings.

Syntax `:HCOPy?`

:HCOPy:ABORt

Function Stops data output and paper feeding.

Syntax `:HCOPy:ABORt`

Example
```
:HCOPY:ABORT
```

:HCOPy:DIRection

Function Sets or queries the data output destination.

Syntax
```
:HCOPy:DIRection {EXTPrinter|PRINter
|NETPrint|FILE|MULTitarget}
:HCOPy:DIRection?
```

Example
```
:HCOPY:DIRECTION EXTPRINTER
:HCOPY:DIRECTION?
-> :HCOPY:DIRECTION EXTPRINTER
```

:HCOPy:EXECute

Function Executes data output.

Syntax `:HCOPy:EXECute`

Example
```
:HCOPY:EXECUTE
```

:HCOPy:EXTPrinter?

Function Queries all external printer output settings.

Syntax `:HCOPy:EXTPrinter?`

:HCOPy:EXTPrinter:MODE

Function Sets or queries the normal or hard copy setting for outputting to the external printer.

Syntax
```
:HCOPy:EXTPrinter:MODE {HARD|NORMal}
:HCOPy:EXTPrinter:MODE?
```

Example
```
:HCOPY:EXTPRINTER:MODE HARD
:HCOPY:EXTPRINTER:MODE?
-> :HCOPY:EXTPRINTER:MODE HARD
```

:HCOPy:EXTPrinter:TONE

Function Sets or queries the external printer output color.

Syntax
```
:HCOPy:EXTPrinter:TONE {<Boolean>}
:HCOPy:EXTPrinter:TONE?
```

Example
```
:HCOPY:EXTPRINTER:TONE ON
:HCOPY:EXTPRINTER:TONE?
-> :HCOPY:EXTPRINTER:TONE 1
```

:HCOPy:MULTitarget?

Function Queries all multi target feature settings.

Syntax `:HCOPy:MULTitarget?`

:HCOPy:MULTitarget:EXTPrinter

Function Sets or queries whether the multi target feature will send output to the external printer.

Syntax
```
:HCOPy:MULTitarget:
EXTPrinter {<Boolean>}
:HCOPy:MULTitarget:EXTPrinter?
```

Example
```
:HCOPY:MULTITARGET:EXTPRINTER ON
:HCOPY:MULTITARGET:EXTPRINTER?
-> :HCOPY:MULTITARGET:EXTPRINTER 1
```

:HCOPy:MULTitarget:PRINter

Function Sets or queries whether the multi target feature will send output to the built-in printer.

Syntax
```
:HCOPy:MULTitarget:
PRINter {<Boolean>}
:HCOPy:MULTitarget:PRINter?
```

Example
```
:HCOPY:MULTITARGET:PRINTER ON
:HCOPY:MULTITARGET:PRINTER?
-> :HCOPY:MULTITARGET:PRINTER 1
```

:HCOPy:MULTitarget:NETPrinter

Function Sets or queries whether the multi target feature will send output to a network printer.

Syntax
```
:HCOPy:MULTitarget:
NETPrinter {<Boolean>}
:HCOPy:MULTitarget:NETPrinter?
```

Example
```
:HCOPY:MULTITARGET:NETPRINTER ON
:HCOPY:MULTITARGET:NETPRINTER?
-> :HCOPY:MULTITARGET:NETPRINTER 1
```

:HCOPy:MULTitarget:FILE

Function Sets or queries whether the multi target feature will send output to a file.

Syntax
```
:HCOPy:MULTitarget:FILE {<Boolean>}
:HCOPy:MULTitarget:FILE?
```

Example
```
:HCOPY:MULTITARGET:FILE ON
:HCOPY:MULTITARGET:FILE?
-> :HCOPY:MULTITARGET:FILE 1
```

:HCOPy:MULTitarget:WAVeform

Function Sets or queries whether the multi target feature will send output to a waveform file.

Syntax
```
:HCOPy:MULTitarget:WAVeform{<Boole
an>}
:HCOPy:MULTitarget:WAVeform?
```

Example
```
:HCOPY:MULTITARGET:WAVEFORM ON
:HCOPY:MULTITARGET:WAVEFORM?
-> :HCOPY:MULTITARGET:WAVEFORM 1
```

:HCOPy:NETPrint?

Function Queries all network printer output settings.

Syntax
```
:HCOPy:NETPrint?
```

:HCOPy:NETPrint:MODE

Function Sets or queries the normal or hard copy setting for outputting to the network printer.

Syntax
```
:HCOPy:NETPrint:MODE {HARD|NORMal}
:HCOPy:NETPrint:MODE?
```

Example
```
:HCOPY:NETPRINT:MODE HARD
:HCOPY:NETPRINT:MODE?
-> :HCOPY:NETPRINT:MODE HARD
```

:HCOPy:NETPrint:TONE

Function Sets or queries the network printer output color.

Syntax
```
:HCOPy:NETPrint:TONE {<Boolean>}
:HCOPy:NETPrint:TONE?
```

Example
```
:HCOPY:NETPRINT:TONE ON
:HCOPY:NETPRINT:TONE?
-> :HCOPY:NETPRINT:TONE 1
```

:HCOPy:NETPrint:TYPE

Function Sets or queries the network printer output command type.

Syntax
```
:HCOPy:NETPrint:TYPE {HINKjet|
HLASer}
:HCOPy:NETPrint:TYPE?
```

Example
```
:HCOPY:NETPRINT:TYPE EINKJET
:HCOPY:NETPRINT:TYPE?
-> :HCOPY:NETPRINT:TYPE EINKJET
```

:HCOPy:PRINter?

Function Queries all built-in printer output settings.

Syntax
```
:HCOPy:PRINter?
```

:HCOPy:PRINter:MODE

Function Sets or queries the short or hard copy setting for outputting to the built-in printer.

Syntax
```
:HCOPy:PRINter:MODE {HARD|SHORt}
:HCOPy:PRINter:MODE?
```

Example
```
:HCOPY:PRINTER:MODE HARD
:HCOPY:PRINTER:MODE?
-> :HCOPY:PRINTER:MODE HARD
```

:HCOPy:PRINter:REPort

Function Sets or queries whether additional information is to be output to the built-in printer.

Syntax
```
:HCOPy:PRINter:REPort {<Boolean>}
:HCOPy:PRINter:REPort?
```

Example
```
:HCOPY:PRINTER:REPORT ON
:HCOPY:PRINTER:REPORT?
-> :HCOPY:PRINTER:REPORT 1
```

5.15 HISTory Group

:HISTory?
Function Queries all of the settings for the history feature.
Syntax :HISTory?

:HISTory:AVERage
Function Sets or queries the history waveform highlight display mode.
Syntax :HISTory:AVERage {<Boolean>}
:HISTory:AVERage?
Example :HISTORY:AVERAGE ON
:HISTORY:AVERAGE?
-> :HISTORY:AVERAGE 1

:HISTory:DISPlay
Function Sets or queries the history record start and end numbers that will be displayed.
Syntax :HISTory:DISPlay {<NRf>,<NRf>}
:HISTory:DISPlay?
<NRf> = See the Features Guide for this information.
Example :HISTORY:DISPLAY 0,-100
:HISTORY:DISPLAY?
-> :HISTORY:DISPLAY 0,-100

:HISTory:DMODe
Function Sets or queries the history waveform display mode.
Syntax :HISTory:DMODe {ALL|COLor|INTensity|ONE}
:HISTory:DMODe?
Example :HISTORY:DMODE ALL
:HISTORY:DMODE?
-> :HISTORY:DMODE ALL

:HISTory:RECord
Function Sets or queries the history waveform source record.
Syntax :HISTory:RECord {<NRf>|MINimum}
:HISTory:RECord?
<NRf> = See the Features Guide for this information.
Example :HISTORY:RECORD 0
:HISTORY:RECORD?
-> :HISTORY:RECORD 0
Description Specify “MINimum” to specify the minimum record number.

:HISTory:RECord? MINimum
Function Queries the smallest record number of the history waveform.
Syntax :HISTory:RECord? {MINimum}
Example :HISTORY:RECORD? MINIMUM
-> :HISTORY:RECORD -1

:HISTory:REPLay?
Function Queries all history waveform replay settings.
Syntax :HISTory:REPLay?

:HISTory:REPLay:JUMP
Function Moves to the specified record number in the history waveform.
Syntax :HISTory:REPLay:JUMP {MAXimum|MINimum}
Example :HISTORY:REPLAY:JUMP MAXIMUM

:HISTory:REPLay:SPEed
Function Sets or queries the history waveform replay speed.
Syntax :HISTory:REPLay:SPEed {<NRf>|PER3|PER10|PER30|PER60}
:HISTory:REPLay:SPEed?
<NRf> = 1, 3, 10
Example :HISTORY:REPLAY:SPEED 1
:HISTORY:REPLAY:SPEED?
-> :HISTORY:REPLAY:SPEED 1

:HISTory:REPLay:STARt
Function Starts replaying the history waveform in the specified direction.
Syntax :HISTory:REPLay:STARt {MAXimum|MINimum}
Example :HISTORY:REPLAY:START MAXIMUM

:HISTory:REPLay:STOP
Function Stops replaying the history waveform.
Syntax :HISTory:REPLay:STOP
Example :HISTORY:REPLAY:STOP

:HISTory[:SEARch]?
Function Queries all history waveform search settings.
Syntax :HISTory[:SEARch]?

:HISTory[:SEARch]:ABORt
Function Aborts the search.
Syntax :HISTory[:SEARch]:ABORt
Example :HISTORY:SEARCH:ABORT

:HISTory[:SEARch]:EXECute
Function Executes a search. This is an overlap command.
Syntax :HISTory[:SEARch]:EXECute
Example :HISTORY:SEARCH:EXECUTE

:HISTory[:SEARch]:LOGic
Function Sets or queries the history waveform search logic.
Syntax :HISTory[:SEARch]:LOGic {AND|OR|SIMPle}
:HISTory[:SEARch]:LOGic?
Example :HISTORY:SEARCH:LOGIC AND
:HISTORY:SEARCH:LOGIC?
-> :HISTORY:SEARCH:LOGIC AND

:HISTory[:SEARch]:NUMBer<x>?
Function Queries all search condition settings.
Syntax :HISTory[:SEARch]:NUMBer<x>?
<x> = 1 to 4

:HISTory[:SEARch]:NUMBer<x>:CONDition
Function Sets or queries the reference condition of the specified search condition.
Syntax :HISTory[:SEARch]:NUMBer<x>:CONDition {IN|OUT}
:HISTory[:SEARch]:NUMBer<x>:CONDition?
<x> = 1 to 4
Example :HISTORY:SEARCH:NUMBER1:CONDITION IN
:HISTORY:SEARCH:NUMBER1:CONDITION?
-> :HISTORY:SEARCH:NUMBER1:
CONDITION IN

:HISTory[:SEARch]:NUMBer<x>:MODE
Function Sets or queries the mode of the specified search condition.
Syntax :HISTory[:SEARch]:NUMBer<x>:MODE {OFF|PARameter|POLYgon|RECTangle|WAVE}
:HISTory[:SEARch]:NUMBer<x>:MODE?
<x> = 1 to 4
Example :HISTORY:SEARCH:NUMBER1:
MODE PARAMETER
:HISTORY:SEARCH:NUMBER1:MODE?
-> :HISTORY:SEARCH:NUMBER1:
MODE PARAMETER

:HISTory[:SEARch]:NUMBer<x>:PARameter?
Function Queries all parameter search settings.
Syntax :HISTory[:SEARch]:NUMBer<x>:PARameter?
<x> = 1 to 4

:HISTory[:SEARch]:NUMBer<x>:PARameter:ITEM
Function Sets or queries the specified waveform parameter's item for parameter searching.
Syntax :HISTory[:SEARch]:NUMBer<x>:PARameter:ITEM {Parameter}
:HISTory[:SEARch]:NUMBer<x>:PARameter:ITEM?
<x> = 1 to 4
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}
Example :HISTORY:SEARCH:NUMBER1:PARAMETER:ITEM AVERAGE
:HISTORY:SEARCH:NUMBER1:PARAMETER:ITEM?
-> :HISTORY:SEARCH:NUMBER1:PARAMETER
:ITEM AVERAGE

:HISTory[:SEARch]:NUMBer<x>:PARameter:LIMit
Function Sets or queries the specified waveform parameter's upper and lower limits for parameter searching.
Syntax :HISTory[:SEARch]:NUMBer<x>:PARameter:LIMit {<NRf>}
:HISTory[:SEARch]:NUMBer<x>:PARameter:LIMit?
<x> = 1 to 4
<NRf> = See the Features Guide for this information.
Example :HISTORY:SEARCH:NUMBER1:PARAMETER:LIMIT 0,1
:HISTORY:SEARCH:NUMBER1:PARAMETER:LIMIT?
-> :HISTORY:SEARCH:NUMBER1:PARAMETER
:LIMIT 1.000E+00,0.000E+00

:HISTory[:SEARch]:NUMBer<x>:PARameter:TRACe

Function Sets or queries the specified waveform parameter's source waveform for parameter searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:PARameter:TRACe {<NRf>|BIT<y>|FFT<y>|MATH<y>|XY<y>}
:HISTory[:SEARch]:NUMBer<x>:PARameter:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of BIT<y> = 1 to 8
<y> of FFT<y> = 1 or 2 (1 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of XY<y> = 1 or 2 (1 on 2-channel models)

Example
```
:HISTORY:SEARCH:NUMBER1:PARAMETER:TRACE 1
:HISTORY:SEARCH:NUMBER1:PARAMETER:TRACE?
-> :HISTORY:SEARCH:NUMBER1:PARAMETER:TRACE 1
```

:HISTory[:SEARch]:NUMBer<x>:PARameter:VALue?

Function Queries the measured value of the specified waveform parameter for parameter searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:PARameter:VALue?
```
<x> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:PARAMETER:VALUE?
-> :HISTORY:SEARCH:NUMBER1:PARAMETER:VALUE 1.98E-03
```

Description If the mode is off or the value cannot be measured, the instrument returns “NAN” (not a number).

:HISTory[:SEARch]:NUMBer<x>:POLYgon?

Function Queries all polygonal zone search settings.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon?
```
<x> = 1 to 4

:HISTory[:SEARch]:NUMBer<x>:POLYgon:HPOSition

Function Sets or queries the horizontal position used in polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:HPOSition {<NRf>}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:HPOSition?
```
<x> = 1 to 4
<NRf> = –5 div to 5 div

Example
```
:HISTORY:SEARCH:NUMBER1:POLYGON:HPOSITION 1
:HISTORY:SEARCH:NUMBER1:POLYGON:HPOSITION?
-> :HISTORY:SEARCH:NUMBER1:POLYGON:HPOSITION 1.000E+00
```

Description When TRACe is XY<x>, <NRf> = –4 div to 4 div.

:HISTory[:SEARch]:NUMBer<x>:POLYgon:RANGe

Function Sets or queries the source window used in polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:RANGe {MAIN|Z1|Z2}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:RANGe?
```
<x> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:POLYGON:RANGE MAIN
:HISTORY:SEARCH:NUMBER1:POLYGON:RANGE?
-> :HISTORY:SEARCH:NUMBER1:POLYGON:RANGE MAIN
```

:HISTory[:SEARch]:NUMBer<x>:POLYgon:TRACe

Function Sets or queries the source waveform used in polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:TRACe {<NRf>|MATH<y>|XY<y>}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of XY<y> = 1 or 2 (1 on 2-channel models)

Example
```
:HISTORY:SEARCH:NUMBER1:POLYGON:TRACE 1
:HISTORY:SEARCH:NUMBER1:POLYGON:TRACE?
-> :HISTORY:SEARCH:NUMBER1:POLYGON:TRACE 1
```

:HISTory[:SEARch]:NUMBer<x>:POLYgon:VPOSition

Function Sets or queries the vertical position used in polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:VPOSition {<NRf>}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:VPOSition?
```
<x> = 1 to 4
<NRf> = –4 div to 4 div

Example
```
:HISTORY:SEARCH:NUMBER1:POLYGON:VPOSITION 1
:HISTORY:SEARCH:NUMBER1:POLYGON:VPOSITION?
-> :HISTORY:SEARCH:NUMBER1:POLYGON:VPOSITION 1.000E+00
```

:HISTory[:SEARch]:NUMBer<x>:POLYgon:ZNUMber

Function Sets or queries the zone number used in polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:ZNUMber {<NRf>}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:ZNUMber?
```
<x> = 1 to 4
<NRf> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:POLYGON:ZNUMBER 1
:HISTORY:SEARCH:NUMBER1:POLYGON:ZNUMBER?
-> :HISTORY:SEARCH:NUMBER1:POLYGON:ZNUMBER 1
```

:HISTory[:SEARch]:NUMBer<x>:RECTangle?

Function Queries all waveform rectangular zone search settings.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:RECTangle?
```
<x> = 1 to 4

:HISTory[:SEARch]:NUMBer<x>:RECTangle:HORizontal

Function Sets or queries the horizontal position of the rectangle used in rectangular zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:RECTangle:HORizontal {<NRf>,<NRf>}
:HISTory[:SEARch]:NUMBer<x>:RECTangle:HORizontal?
```
<x> = 1 to 4
<NRf> = –5 div to 5 div

Example
```
:HISTORY:SEARCH:NUMBER1:RECTANGLE:HORIZONTAL 1,2
:HISTORY:SEARCH:NUMBER1:RECTANGLE:HORIZONTAL?
-> :HISTORY:SEARCH:NUMBER1:RECTANGLE:HORIZONTAL 2.000E+00,1.000E+00
```

Description When TRACe is XY<x>, <NRf> = –4 div to 4 div.

:HISTory[:SEARch]:NUMBer<x>:RECTangle:RANGe

Function Sets or queries the source window of the rectangle used in rectangular zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:RECTangle:RANGe {MAIN|Z1|Z2}
:HISTory[:SEARch]:NUMBer<x>:RECTangle:RANGe?
```
<x> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:RECTANGLE:RANGE MAIN
:HISTORY:SEARCH:NUMBER1:RECTANGLE:RANGE?
-> :HISTORY:SEARCH:NUMBER1:RECTANGLE:RANGE MAIN
```

:HISTory[:SEARch]:NUMBer<x>:RECTangle:TRACe

Function Sets or queries the source waveform of the rectangle used in rectangular zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:RECTangle:TRACe {<NRf>|MATH<y>|XY<y>}
:HISTory[:SEARch]:NUMBer<x>:RECTangle:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of XY<y> = 1 or 2 (1 on 2-channel models)

Example
```
:HISTORY:SEARCH:NUMBER1:RECTANGLE:TRACE 1
:HISTORY:SEARCH:NUMBER1:RECTANGLE:TRACE?
-> :HISTORY:SEARCH:NUMBER1:RECTANGLE:TRACE 1
```

:HISTory[:SEARch]:NUMBer<x>:RECTangle:VERTical

Function Sets or queries the vertical position of the rectangle used in rectangular zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:RECTangle:VERTical {<NRf>,<NRf>}
:HISTory[:SEARch]:NUMBer<x>:RECTangle:VERTical?
```
<x> = 1 to 4
<NRf> = –4 div to 4 div

Example
```
:HISTORY:SEARCH:NUMBER1:RECTANGLE:VERTICAL 1,2
:HISTORY:SEARCH:NUMBER1:RECTANGLE:VERTICAL?
-> :HISTORY:SEARCH:NUMBER1:RECTANGLE:VERTICAL 2.000E+00,1.000E+00
```

:HISTory[:SEARch]:NUMBer<x>:WAVE?

Function Queries all waveform waveform zone search settings.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE?
```
<x> = 1 to 4

:HISTory[:SEARch]:NUMBer<x>:WAVE:RANGe

Function Sets or queries the source window used in waveform zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE:RANGe {MAIN|Z1|Z2}
:HISTory[:SEARch]:NUMBer<x>:WAVE:RANGe?
```
<x> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:WAVE:RANGE MAIN
:HISTORY:SEARCH:NUMBER1:WAVE:RANGE?
-> :HISTORY:SEARCH:NUMBER1:WAVE:RANGE MAIN
```

:HISTory[:SEARch]:NUMBer<x>:WAVE:TRACe

Function Sets or queries the source waveform used in waveform zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRACe {<NRf>|MATH<y>}
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRACe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:HISTORY:SEARCH:NUMBER1:WAVE:TRACE 1
:HISTORY:SEARCH:NUMBER1:WAVE:TRACE?
-> :HISTORY:SEARCH:NUMBER1:WAVE:TRACE 1
```

:HISTory[:SEARch]:NUMBer<x>:WAVE:TRANge

Function Sets or queries the determination period used in waveform zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRANge {<NRf>,<NRf>}
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRANge?
```
<NRf> = –5 div to 5 div

Example
```
:HISTORY:SEARCH:NUMBER1:WAVE:TRANGE 1,2
:HISTORY:SEARCH:NUMBER1:WAVE:TRANGE?
-> :HISTORY:SEARCH:NUMBER1:WAVE:TRANGE 2.000E+00,1.000E+00
```

:HISTory[:SEARch]:NUMBer<x>:WAVE:ZNUMber

Function Sets or queries the zone number used in waveform zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE:ZNUMber {<NRf>}
:HISTory[:SEARch]:NUMBer<x>:WAVE:ZNUMber?
```
<x> = 1 to 4
<NRf> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:WAVE:ZNUMBER 1
:HISTORY:SEARCH:NUMBER1:WAVE:ZNUMBER?
-> :HISTORY:SEARCH:NUMBER1:WAVE:ZNUMBER 1
```

:HISTory[:SEARch]:SIMPle?

Function Queries all simple searching settings.

Syntax
```
:HISTory[:SEARch]:SIMPle?
```

:HISTory[:SEARch]:SIMPle:HORizontal

Function Sets or queries the horizontal position of the rectangle to use in simple searching.

Syntax
```
:HISTory[:SEARch]:SIMPle:HORizontal {<NRf>,<NRf>}
:HISTory[:SEARch]:SIMPle:HORizontal?
```
<NRf> = –5 div to 5 div

Example
```
:HISTORY:SEARCH:SIMPLE:HORIZONTAL 1,2
:HISTORY:SEARCH:SIMPLE:HORIZONTAL?
-> :HISTORY:SEARCH:SIMPLE:HORIZONTAL 2.000E+00,1.000E+00
```

:HISTory[:SEARch]:SIMPle:RANGe

Function Sets or queries the target window of the rectangle to use in simple searching.

Syntax
```
:HISTory[:SEARch]:SIMPle:RANGe {MAIN|Z1|Z2}
:HISTory[:SEARch]:SIMPle:RANGe?
```

Example
```
:HISTORY:SEARCH:SIMPLE:RANGE MAIN
:HISTORY:SEARCH:SIMPLE:RANGE?
-> :HISTORY:SEARCH:SIMPLE:RANGE MAIN
```

:HISTory[:SEARch]:SIMPle:TRACe

Function Sets or queries the source trace of the rectangle to use in simple searching.

Syntax
```
:HISTory[:SEARch]:SIMPle:TRACe {<NRf>|MATH<x>|XY<x>}
:HISTory[:SEARch]:SIMPle:TRACe?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of XY<x> = 1 or 2 (1 on 2-channel models)

Example
```
:HISTORY:SEARCH:SIMPLE:TRACE 1
:HISTORY:SEARCH:SIMPLE:TRACE?
-> :HISTORY:SEARCH:SIMPLE:TRACE 1
```

:HISTory[:SEARch]:SIMPle:VERTical

Function Sets or queries the vertical position of the rectangle to use in simple searching.

Syntax
```
:HISTory[:SEARch]:SIMPle:VERTical {<NRf>,<NRf>}
:HISTory[:SEARch]:SIMPle:VERTical?
```
<NRf> = -4 div to 4 div

Example
```
:HISTORY:SEARCH:SIMPLE:VERTICAL 1,2
:HISTORY:SEARCH:SIMPLE:VERTICAL?
-> :HISTORY:SEARCH:SIMPLE:VERTICAL 2.000E+00,1.000E+00
```

:HISTory:TIME?

Function Queries the time of the source record number.

Syntax
```
:HISTory:TIME? {<NRf>|MINimum}
```

Example
```
:HISTORY:TIME? -100
-> :HISTORY:TIME "-100 10:20:30.400"
```

Description Specify “MINimum” to specify the minimum record number.

5.16 IMAGe Group

:IMAGe?

Function Queries all screen capture data output settings.
Syntax :IMAGe?

:IMAGe:ABORt

Function Aborts saving screen capture data to the storage device.
Syntax :IMAGe:ABORt
Example :IMAGE:ABORT

:IMAGe:BACKground

Function Sets or queries the screen capture background.
Syntax :IMAGe:BACKground {NORMal|TRANsparent}
:IMAGe:BACKground?
Example :IMAGE:BACKGROUND NORMAL
:IMAGE:BACKGROUND?
-> :IMAGE:BACKGROUND NORMAL

:IMAGe:EXECute

Function Saves the screen capture data to the storage device.
Syntax :IMAGe:EXECute
:IMAGE:EXECUTE

:IMAGe:FORMat

Function Sets or queries the screen capture output format.
Syntax :IMAGe:FORMat {BMP|JPEG|PNG}
:IMAGe:FORMat?
Example :IMAGE:FORMAT BMP
:IMAGE:FORMAT?
-> :IMAGE:FORMAT BMP

:IMAGe:INFormation

Function Sets or queries whether setting information is included in screen capture data.
Syntax :IMAGe:INFormation {<Boolean>}
:IMAGe:INFormation?
Example :IMAGE:INFORMATION ON
:IMAGE:INFORMATION?
-> :IMAGE:INFORMATION 1

:IMAGe:MODE

Function Sets or queries the screen capture output mode.
Syntax :IMAGe:MODE {HARD|NORMal|WIDE}
:IMAGe:MODE?
Example :IMAGE:MODE HARD
:IMAGE:MODE?
-> :IMAGE:MODE HARD

:IMAGe:SAVE?

Function Queries all file output settings.
Syntax :IMAGe:SAVE?

:IMAGe:SAVE:ANAMing

Function Sets or queries the on/off state of the auto naming feature for saving files.
Syntax :IMAGe:SAVE:ANAMing {DATE|DATE2|NUMBering|OFF}
:IMAGe:SAVE:ANAMing?
Example :IMAGE:SAVE:ANAMING DATE
:IMAGE:SAVE:ANAMING?
-> :IMAGE:SAVE:ANAMING DATE

:IMAGe:SAVE:CDIRectory

Function Changes the file directory.
Syntax :IMAGe:SAVE:CDIRectory {<String>}
<String> = See the Features Guide for this information.
Example :IMAGE:SAVE:CDIRECTORY "ABC"

:IMAGe:SAVE:DRIVe

Function Sets the storage device that the file will be created in.
Syntax :IMAGe:SAVE:DRIVe {FLAShmem|NETWork|USB,<NRf>}
<NRf> = 0 to 3
Example :IMAGE:SAVE:DRIVE FLASHMEM
Description The <NRf> for USB can be omitted if the drive is not partitioned or divided by LUN.

:IMAGe:SAVE:NAME

Function Sets or queries the name of the file to be created.

Syntax
```
:IMAGe:SAVE:NAME {<Filename>}
:IMAGe:SAVE:NAME?
```

Example
```
:IMAGE:SAVE:NAME "DISP_1"
:IMAGE:SAVE:NAME?
-> :IMAGE:SAVE:NAME "DISP_1"
```

:IMAGe:SEND?

Function Queries the screen capture data value.

Syntax
```
:IMAGe:SEND?
```

Example
```
:IMAGE:SEND?
-> : IMAGE: SEND #8 (8-digit number
of bytes)(data byte sequence)(block
data)
```

Description For details on <block data>, see page 4-7.

:IMAGe:TONE

Function Sets or queries the color tone of the screen capture data that will be saved.

Syntax
```
:IMAGe:TONE {COLor|GRAY|OFF|REVerse}
:IMAGe:TONE?
```

Example
```
:IMAGE:TONE COLOR
:IMAGE:TONE?
-> :IMAGE:TONE COLOR
```

5.17 INITialize Group

:INITialize:EXECute

Function Executes initialization.

Syntax `:INITialize:EXECute`

Example `:INITIALIZE:EXECUTE`

:INITialize:UNDO

Function Undoes the initialization that was executed.

Syntax `:INITialize:UNDO`

Example `:INITIALIZE:UNDO`

5.18 LOGic Group

The commands in this group cannot be used on models without logic input.

:LOGic?

Function Queries all logic input waveform settings.

Syntax
```
:LOGic?
```

:LOGic:MODE

Function Sets or queries the logic input on/off state.

Syntax
```
:LOGic:MODE {<Boolean>}
:LOGic:MODE?
```

Example
```
:LOGIC:MODE ON
:LOGIC:MODE?
-> :LOGIC:MODE 1
```

:LOGic[:PODA]?

Function Queries all logic input settings.

Syntax
```
:LOGic[:PODA]?
```

:LOGic[:PODA]:BITOrder

Function Sets or queries bit order of the logic input.

Syntax
```
:LOGic[:PODA]:BITOrder {<String>}
:LOGic[:PODA]:BITOrder?
```
<String> = Combination of “BIT0” to “BIT7” and “BUS” (35 characters)

Example
```
:LOGIC:PODA:BITORDER "BIT0BIT1BIT2BIT3BIT4BIT5BIT6BIT7BUS"
:LOGIC:PODA:BITORDER?
-> :LOGIC:PODA:BITORDER "BIT0BIT1BIT2BIT3BIT4BIT5BIT6BIT7BUS"
```

:LOGic[:PODA]:BITSetup:ALL:DISPlay

Function Turns on or off all bit displays of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:ALL:DISPlay {<Boolean>}
:LOGic[:PODA]:BITSetup:ALL:DISPlay?
```

Example
```
:LOGIC:PODA:ALL:DISPLAY ON
:LOGIC:PODA:ALL:DISPLAY?
-> :LOGIC:PODA:ALL:DISPLAY 1
```

:LOGic[:PODA]:BITSetup:ALL:LEVel

Function Sets or queries the user-defined threshold level of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:ALL:LEVel {<Voltage>}
:LOGic[:PODA]:BITSetup:ALL:LEVel?
```
<Voltage> = –10 to 10 V (0.1 V steps)

Example
```
:LOGIC:PODA:BITSETUP:ALL:LEVEL 1V
:LOGIC:PODA:BITSETUP:ALL:LEVEL?
-> :LOGIC:PODA:BITSETUP:ALL:LEVEL 1.0E+00
```

:LOGic[:PODA]:BITSetup:ALL:TYPE

Function Selects the threshold level of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:ALL:TYPE {CMOS1|CMOS2|CMOS3|CMOS5|ECL}
```

Example
```
:LOGIC:PODA:BITSETUP:ALL:TYPE CMOS1
```

:LOGic[:PODA]:BITSetup:BIT<x>?

Function Queries all settings of a bit of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:BIT<x>?
```
<x> = 1 to 8

:LOGic[:PODA]:BITSetup:BIT<x>:DISPlay

Function Turns on or off all bit displays of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:BIT<x>:DISPlay {<Boolean>}
:LOGic[:PODA]:BITSetup:BIT<x>:DISPlay?
```
<x> = 1 to 8

Example
```
:LOGIC:PODA:BITSETUP:BIT1:DISPLAY ON
:LOGIC:PODA:BITSETUP:BIT1:DISPLAY?
-> :LOGIC:PODA:BITSETUP:BIT1:DISPLAY 1
```

:LOGic[:PODA]:BITSetup:BIT<x>:LABel

Function Sets or queries the label of a bit of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:BIT<x>:
LABel {<String>}
:LOGic[:PODA]:BITSetup:BIT<x>:LABel?
```
<x> = 1 to 8
<String> = Up to 8 characters

Example
```
:LOGIC:PODA:BITSETUP:BIT1:
LABEL "NO_1"
:LOGIC:PODA:BITSETUP:BIT1:LABEL?
-> :LOGIC:PODA:BITSETUP:BIT1:
LABEL "NO_1"
```

:LOGic[:PODA]:BITSetup:BIT<x>:LEVel

Function Sets or queries the user-defined threshold level of a bit of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:BIT<x>:
LEVel {<Voltage>}
:LOGic[:PODA]:BITSetup:BIT<x>:LEVel?
```
<x> = 1 to 8
<Voltage> = –10 to 10 V (0.1 V steps)

Example
```
:LOGIC:PODA:BITSETUP:BIT1:LEVEL 1V
:LOGIC:PODA:BITSETUP:BIT1:LEVEL?
-> :LOGIC:PODA:BITSETUP:BIT1:
LEVEL 1.0E+00
```

:LOGic[:PODA]:BITSetup:BIT<x>:TYPE

Function Selects the threshold level of a bit of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:BITSetup:BIT<x>:TY
PE {CMOS1|CMOS2|CMOS3|CMOS5|ECL}
```
<x> = 1 to 8

Example
```
:LOGIC:PODA:BITSETUP:BIT1:TYPE CMOS1
```

:LOGic[:PODA]:BUS?

Function Queries all logic input bus settings.

Syntax
```
:LOGic[:PODA]:BUS?
```

Example
```
:LOGIC:PODA:BUS?
```

:LOGic[:PODA]:BUS:ASSignment

Function Sets or queries the logic input bus assignment.

Syntax
```
:LOGic[:PODA]:BUS:ASSignment {<String>
}
:LOGic[:PODA]:BUS:ASSignment?
```
<String> = Up to 32 characters

Example
```
:LOGIC:PODA:BUS:ASSIGMMENT "BIT1"
:LOGIC:PODA:BUS:ASSIGMMENT?
-> :LOGIC:PODA:BUS:ASSIGMMENT "BIT1"
```

:LOGic[:PODA]:BUS:DISPlay

Function Sets or queries the on/off state of the logic input bus display.

Syntax
```
:LOGic[:PODA]:BUS:DISPlay {<Boolean>
}
:LOGic[:PODA]:BUS:DISPlay?
```

Example
```
:LOGIC:PODA:BUS:DISPLAY ON
:LOGIC:PODA:BUS:DISPLAY?
-> :LOGIC:PODA:BUS:DISPLAY ON
```

:LOGic[:PODA]:BUS:FORMat

Function Sets or queries the logic input bus display format.

Syntax
```
:LOGic[:PODA]:BUS:FORMat {BINary|
HEX}
:LOGic[:PODA]:BUS:FORMat?
```

Example
```
:LOGIC:PODA:BUS:FORMAT HEX
:LOGIC:PODA:BUS:FORMAT?
-> :LOGIC:PODA:BUS:FORMAT HEX
```

:LOGic[:PODA]:BUS:LABel

Function Sets or queries the label of the logic input bus label.

Syntax
```
:LOGic[:PODA]:BUS:LABel {<String>}
LOGic[:PODA]:BUS:LABel?
```
<String> = Up to 8 characters

Example
```
:LOGIC:PODA:BUS:LABEL "LOGIC"
:LOGIC:PODA:BUS:LABEL?
-> :LOGIC:PODA:BUS:LABEL "LOGIC"
```

:LOGic[:PODA]:DESKew

Function Sets or queries the deskewing of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:DESKew {<Time>}
:LOGic[:PODA]:DESKew?
```
<Time> = –1000 ns to 1000 ns (10 ps steps)

Example
```
:LOGIC:PODA:DESKEW 1NS
:LOGIC:PODA:DESKEW?
-> :LOGIC:PODA:DESKEW 1.000E-09
```

:LOGic[:PODA]:HYSTeresis

Function Sets or queries the hysteresis of the specified pod of the logic input.

Syntax
```
:LOGic[:PODA]:HYSTeresis {HIGH|LOW}
:LOGic[:PODA]:HYSTeresis?
```

Example
```
:LOGIC:PODA:HYSTERESIS HIGH
:LOGIC:PODA:HYSTERESIS?
-> :LOGIC:PODA:HYSTERESIS HIGH
```

:LOGic[:PODA]:STATe:ASSignment?

Function Queries the logic input state display assignment.

Syntax
```
:LOGic[:PODA]:STATe:ASSignment?
```

Example
```
:LOGIC:PODA:STATE:ASSIGMMENT?
```

:LOGic[:PODA]:STATe:ASSignment:ALL

Function Sets or queries all settings related to the logic input state display assignment.

Syntax
```
:LOGic[:PODA]:STATe:ASSignment:A
LL {<Boolean>}
:LOGic[:PODA]:STATe:ASSignment:ALL?
```

Example
```
:LOGIC:PODA:STATE:ASSIGMMENT:ALL ON
:LOGIC:PODA:STATE:ASSIGMMENT:ALL?
-> :LOGIC:PODA:STATE:ASSIGMMENT:A
LL 1
```

:LOGic[:PODA]:STATe:ASSignment:BIT<x>

Function Sets or queries the logic input state display assignments of all bits.

Syntax
```
:LOGic[:PODA]:STATe:ASSignment:BIT
<x> {<Boolean>}
:LOGic[:PODA]:STATe:ASSignment:BIT
<x>?
```
<x> = 1 to 8

Example
```
:LOGIC:PODA:STATE:ASSIGMMENT:BIT1 ON
:LOGIC:PODA:STATE:ASSIGMMENT:BIT1?
-> :LOGIC:PODA:STATE:ASSIGMMENT:B
IT1 1
```

:LOGic[:PODA]:STATe:ASSignment:BUS

Function Sets or queries the bus setting of the logic input state display assignment.

Syntax
```
:LOGic[:PODA]:STATe:ASSignment:B
US {<Boolean>}
:LOGic[:PODA]:STATe:ASSignment:BUS?
```

Example
```
:LOGIC:PODA:STATE:ASSIGMMENT:BUS ON
:LOGIC:PODA:STATE:ASSIGMMENT:BUS?
-> :LOGIC:PODA:STATE:ASSIGMMENT:B
US 1
```

:LOGic:[:PODA]:STATe:CLOCk

Function Sets or queries the state display reference clock waveform of the logic input.

Syntax
```
:LOGic:[:PODA]:STATe:CLOCk {<NRf>|
BIT<x>}
:LOGic[:PODA:]STATe:CLOCk?
```
<NRf> = 1 to 3
<x> = 1 to 8

Example
```
:LOGIC:PODA:STATE:CLOCK 1
:LOGIC:PODA:STATE:CLOCK?
-> :LOGIC:PODA:STATE:CLOCK 1
```

:LOGic:[:PODA]:STATe:HYSTeresis

Function Sets or queries the hysteresis of the state display reference clock waveform of the logic input.

Syntax
```
:LOGic:[:PODA]:STATe:
HYSTeresis {<NRf>}
:LOGic[:PODA]:STATe:HYSTeresis?
```
<NRf> = 0.3 to 8.0 (div)

Example
```
:LOGIC:PODA:STATE:HYSTERESIS 1.0
:LOGIC:PODA:STATE:HYSTERESIS?
-> :LOGIC:PODA:STATE:
HYSTERESIS 1.000E+00
```

:LOGic:[:PODA]:STATe:MODE

Function Sets or queries the on/off state of the logic input state display.

Syntax
```
:LOGic:[:PODA]:STATe:
MODE {<Boolean>}
:LOGic[:PODA]:STATe:MODE?
```

Example
```
:LOGIC:PODA:STATE:MODE ON
:LOGIC:PODA:STATE:MODE?
-> :LOGIC:PODA:STATE:MODE 1
```

:LOGic:[:PODA]:STATe:POLarity

Function Sets or queries the polarity of the state display reference clock waveform of the logic input.

Syntax
```
:LOGic:[:PODA]:STATe:POLarity {RISE|
FALL|BOTH}
:LOGic[:PODA]:STATe:POLarity?
```

Example
```
:LOGIC:PODA:STATE:POLARITY RISE
:LOGIC:PODA:STATE:POLARITY?
-> :LOGIC:PODA:STATE:POLARITY RISE
```

:LOGic:[:PODA]:STATe:THReshold

Function Sets or queries the detection level of the state display reference clock waveform of the logic input.

Syntax
```
:LOGic:[:PODA]:STATe:
THReshold {<NRf>}
:LOGic[:PODA]:STATe:THReshold?
```
<NRf> = See the Features Guide for this information.

Example
```
:LOGIC:PODA:STATE:THRESHOLD 1
:LOGIC:PODA:STATE:THRESHOLD?
-> :LOGIC:PODA:STATE:
THRESHOLD 1.000E+00
```

:LOGic:POSition

Function Sets or queries the vertical position of the logic signal.

Syntax
```
:LOGic:POSition {<NRf>}
:LOGic:POSition?
```
<NRf> = –27 to 80

Example
```
:LOGIC:POSITION 0
:LOGIC:POSITION?
-> :LOGIC:POSITION 0
```

:LOGic:SIZE

Function Sets or queries the display size of the logic signal.

Syntax
```
:LOGic:SIZE {LARGe|MEDium|SMALl}
:LOGic:SIZE?
```

Example
```
:LOGIC:SIZE LARGE
:LOGIC:SIZE?
-> :LOGIC:SIZE LARGE
```

5.19 MATH Group

User-defined computation is an option available only for 4-channel models. User-defined computation commands cannot be used on 2-channel models or 4-channel models without the user-defined computation option.

:MATH<x>?

Function Queries all computation settings.

Syntax `:MATH<x>?`

<x> = 1 to 4 (1 or 2 on 2-channel models)

:MATH<x>:DISPlay

Function Sets or queries whether to show (ON) or hide (OFF) the computed waveforms.

Syntax `:MATH<x>:DISPlay {<Boolean>}`
`:MATH<x>:DISPlay?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MATH1:DISPLAY ON
:MATH1:DISPLAY?
-> :MATH1:DISPLAY 1
```

:MATH<x>:ECOunt?

Function Queries all edge count settings.

Syntax `:MATH<x>:ECOunt?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

:MATH<x>:ECOunt:HYSTeresis

Function Sets or queries the edge detection level hysteresis for edge count.

Syntax `:MATH<x>:ECOunt:HYSTeresis {<NRf>}`
`:MATH<x>:ECOunt:HYSTeresis?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 0.3 to 8.0 (div)

Example
```
:MATH1:ECOUNT:HYSTERESIS 1
:MATH1:ECOUNT:HYSTERESIS?
-> :MATH1:ECOUNT:
HYSTERESIS 1.000E+00
```

:MATH<x>:ECOunt:POLarity

Function Sets or queries the edge detection polarity for edge count.

Syntax `:MATH<x>:ECOunt:POLarity {FALL|RISE}`
`:MATH<x>:ECOunt:POLarity?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MATH1:ECOUNT:POLARITY FALL
:MATH1:ECOUNT:POLARITY?
-> :MATH1:ECOUNT:POLARITY FALL
```

:MATH<x>:ECOunt:THReshold

Function Sets or queries the edge detection Level for edge count computation.

Syntax `:MATH<x>:ECOunt:THReshold {<NRf>|<Voltage>|<Current>}`
`:MATH<x>:ECOunt:THReshold?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<Voltage>, <Current> = See the Features Guide for this information.

Example
```
:MATH1:ECOUNT:THRESHOLD 1
:MATH1:ECOUNT:THRESHOLD?
-> :MATH1:ECOUNT:THRESHOLD 1.000E+00
```

:MATH<x>:FILTer?

Function Queries all filter settings.

Syntax `:MATH<x>:FILTer?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

:MATH<x>:FILTer:FORDer

Function Sets or queries the IIR filter's filter order.

Syntax `:MATH<x>:FILTer:FORDer {<NRf>}`
`:MATH<x>:FILTer:FORDer?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1 or 2

Example
```
:MATH1:FILTER:FORDER 1
:MATH1:FILTER:FORDER?
-> :MATH1:FILTER:FORDER 1
```

:MATH<x>:FILTer:HCUToff

Function Sets or queries the cutoff frequency of the IIR filter high-pass filter.

Syntax `:MATH<x>:FILTer:HCUToff {<Frequency>}`
`:MATH<x>:FILTer:HCUToff?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<Frequency> = See the Features Guide for this information.

Example
```
:MATH1:FILTER:HCUTOFF 10MHZ
:MATH1:FILTER:HCUTOFF?
-> :MATH1:FILTER:HCUTOFF 10.00E+06
```

:MATH<x>:FILTer:LCUToff

Function Sets or queries the cutoff frequency of the IIR filter low-pass filter.

Syntax
```
:MATH<x>:FILTer:LCUToff {<Frequency>}
:MATH<x>:FILTer:LCUToff?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<Frequency> = See the Features Guide for this information.

Example
```
:MATH1:FILTER:LCUTOFF 10HZ
:MATH1:FILTER:LCUTOFF?
-> :MATH1:FILTER:LCUTOFF 10.00E+00
```

:MATH<x>:FILTer:TIME

Function Sets or queries the phase shift delay.

Syntax
```
:MATH<x>:FILTer:TIME {<Time>}
:MATH<x>:FILTer:TIME?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Time> = See the Features Guide for this information.

Example
```
:MATH1:FILTER:TIME 1S
:MATH1:FILTER:TIME?
-> :MATH1:FILTER:TIME 1.000E+00
```

:MATH<x>:FILTer:TYPE

Function Sets or queries the filter type.

Syntax
```
:MATH<x>:FILTer:TYPE {DELay|MAVG|
IHPass|ILPass}
:MATH<x>:FILTer:TYPE?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MATH1:FILTER:TYPE DELAY
:MATH1:FILTER:TYPE?
-> :MATH1:FILTER:TYPE DELAY
```

:MATH<x>:FILTer:WEIGht

Function Sets or queries the moving average weight.

Syntax
```
:MATH<x>:FILTer:WEIGht {<NRf>}
:MATH<x>:FILTer:WEIGht?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 2 to 128 (in 2n steps)

Example
```
:MATH1:FILTER:WEIGHT 2
:MATH1:FILTER:WEIGHT?
-> :MATH1:FILTER:WEIGHT 2
```

:MATH<x>:INTegral?

Function Queries all integral settings.

Syntax
```
:MATH<x>:INTegral?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

:MATH<x>:INTegral:SPOint

Function Sets or queries the integral start position.

Syntax
```
:MATH<x>:INTegral:SPOint {<NRf>}
:MATH<x>:INTegral:SPOint?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:MATH1:INTEGRAL:SPOINT 1.5
:MATH1:INTEGRAL:SPOINT?
-> :MATH1:INTEGRAL:SPOINT 1.5000000
```

:MATH<x>:LABel?

Function Queries all settings related to the label of the computed waveform.

Syntax
```
:MATH<x>:LABel?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

:MATH<x>:LABel[:DEFine]

Function Sets or queries a computed waveform label.

Syntax
```
:MATH<x>:LABel[:DEFine] {<String>}
:MATH<x>:LABel[:DEFine]?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<String> = Up to 8 characters

Example
```
:MATH1:LABEL:DEFINE "MATH1"
:MATH1:LABEL:DEFINE?
-> :MATH1:LABEL:DEFINE "MATH1"
```

:MATH<x>:LABel:MODE

Function Sets or queries the label display on/off state of computed waveforms.

Syntax
```
:MATH<x>:LABel:MODE {<Boolean>}
:MATH<x>:LABel:MODE?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MATH1:LABEL:MODE ON
:MATH1:LABEL:MODE?
-> :MATH1:LABEL:MODE 1
```

:MATH<x>:OPERation

Function Sets or queries the operator.

Syntax
```
:MATH<x>:OPERation {ECOunt|FILTer|
INTegral|MINus|MULTiple|PLUS|RCOunt|
USERdefine},{<NRf>|MATH<x>}
[,{<NRf>|MATH<x>}]
[,{<NRf>|MATH<x>}|NONE]
:MATH<x>:OPERation?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1 to 4

Example
```
:MATH1:OPERATION PLUS,1,2
:MATH1:OPERATION?
-> :MATH1:OPERATION PLUS,1,2
```

Description
- For unary operators (ECOunt|FILTer|INTegral), select the source waveform with the first <NRf>.
- For binary operators (MINus|MULTiple|PLUS), select the source waveform for the first and second term with the first and second <NRf>, respectively.
- For ternary operators (RCOunt), select the source waveform for the first, second, and third terms with the first, second, and third <NRf>, respectively.
- For the USERdefine operator, <NRf> is not necessary.

:MATH:RCOunt?

Function Queries all rotary count computation settings.

Syntax
```
:MATH:RCOunt?
```

:MATH<x>:RCOunt:HYSTeresis<y>

Function Sets or queries the hysteresis for rotary count computation.

Syntax
```
:MATH<x>:RCOunt:
HYSTeresis<y> {<NRf>}
:MATH<x>:RCOunt:HYSTeresis<y>?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<y> = 1 to 3
<NRf> = 0.3 to 8.0

Example
```
:MATH1:RCOUNT:HYSTERESIS1 0.3
:MATH1:RCOUNT:HYSTERESIS1?
-> :MATH1:RCOUNT:HYSTERESIS1 0.3000
```

:MATH<x>:RCOunt:THReshold<y>

Function Sets or queries the determination level for rotary count computation.

Syntax
```
:MATH<x>:RCOunt:THReshold<y> {<NRf>|
<Voltage>|<Current>}
:MATH<x>:RCOunt:THReshold<y>?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<y> = 1 to 3
<Voltage>, <Current> = See the Features Guide for this information.

Example
```
:MATH1:RCOUNT:THRESHOLD1 1
:MATH1:RCOUNT:THRESHOLD1?
-> :MATH1:RCOUNT:
THRESHOLD1 1.000E+00
```

:MATH<x>:RCOunt:ZPOLarity

Function Sets or queries the polarity for rotary count computation.

Syntax
```
:MATH<x>:RCOunt:ZPOLarity {HIGH|LOW}
:MATH<x>:RCOunt:ZPOLarity?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MATH1:RCOUNT:ZPOLARITY HIGH
:MATH1:RCOUNT:ZPOLARITY?
-> :MATH1:RCOUNT:ZPOLARITY HIGH
```

:MATH<x>:SCALe?

Function Queries all scaling settings.

Syntax
```
:MATH<x>:SCALe?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

:MATH<x>:SCALe:CENTer

Function Sets or queries the center value for manual scaling.

Syntax
```
:MATH<x>:SCALe:CENTer {<NRf>}
:MATH<x>:SCALe:CENTer?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:MATH1:SCALE:CENTER 1
:MATH1:SCALE:CENTER?
-> :MATH1:SCALE:CENTER 1.00000E+00
```

:MATH<x>:SCALe:MODE

Function Sets or queries the scale mode.

Syntax
```
:MATH<x>:SCALe:MODE {AUTO|MANual}
:MATH<x>:SCALe:MODE?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MATH1:SCALE:MODE AUTO
:MATH1:SCALE:MODE?
-> :MATH1:SCALE:MODE AUTO
```

:MATH<x>:SCALe:SENSitivity

Function Sets or queries the sensitivity for manual scaling.

Syntax
```
:MATH<x>:SCALe:SENSitivity {<NRf>}
:MATH<x>:SCALe:SENSitivity?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1.0000E–31 to 1.0000E+31

Example
```
:MATH1:SCALE:SENSITIVITY 10
:MATH1:SCALE:SENSITIVITY?
-> :MATH1:SCALE:
SENSITIVITY 10.0000E+00
```

:MATH<x>:UNIT?

Function Queries all computation unit settings.

Syntax
```
:MATH<x>:UNIT?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

:MATH<x>:UNIT[:DEFine]

Function Sets or queries the computation unit.

Syntax
```
:MATH<x>:UNIT[:DEFine] {<String>}
:MATH<x>:UNIT[:DEFine]?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)
<String> = Up to 4 characters

Example
```
:MATH1:UNIT:DEFINE "EU"
:MATH1:UNIT:DEFINE?
-> :MATH1:UNIT:DEFINE "EU"
```

Description The unit is applied to scaled values. This command has no effect on the computation result.

:MATH<x>:UNIT:MODE

Function Sets or queries whether a computation unit will be attached automatically or manually.

Syntax
```
:MATH<x>:UNIT:MODE {AUTO|USERdefine}
:MATH<x>:UNIT:MODE?
```
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MATH1:UNIT:MODE AUTO
:MATH1:UNIT:MODE?
-> :MATH1:UNIT:MODE AUTO
```

:MATH<x>:USERdefine?

Function Queries all user-defined computation settings.

Syntax
```
:MATH<x>:USERdefine?
```
<x> = 1 to 4

:MATH<x>:USERdefine:AVERage?

Function Queries all settings related to the averaging of user-defined computation.

Syntax
```
:MATH<x>:USERdefine:AVERage?
```
<x> = 1 to 4

Description This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:AVERage:EWEight

Function Sets or queries the attenuation constant of exponential averaging of user-defined computation.

Syntax
```
:MATH<x>:USERdefine:AVERage:EWEig
ht {<NRf>}
:MATH<x>:USERdefine:AVERage:EWEight?
```
<x> = 1 to 4
<NRf> = 2 to 1024 (in 2n steps)

Example
```
:MATH1:USERDEFINE:AVERAGE:EWEIGHT 2
:MATH1:USERDEFINE:AVERAGE:EWEIGHT?
-> :MATH1:USERDEFINE:AVERAGE:
EWEIGHT 2
```

Description This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:AVERage:MODE

Function Sets or queries the averaging mode for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:AVERage:MO
DE {EXPonent|OFF}
:MATH<x>:USERdefine:AVERage:MODE?
```
<x> = 1 to 4

Example
```
:MATH1:USERDEFINE:AVERAGE:
MODE EXPONENT
:MATH1:USERDEFINE:AVERAGE:MODE?
-> :MATH1:USERDEFINE:AVERAGE:
MODE EXPONENT
```

Description This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:CONSitant<y>

Function Sets or queries a constant for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:CONSitant
<y> {<NRf>}
:MATH<x>:USERdefine:CONSitant<y>?
```
<x> = 1 to 4
<y> = 1 to 4
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:MATH1:USERDEFINE:CONSITANT1 1
:MATH1:USERDEFINE:CONSITANT1?
-> :MATH1:USERDEFINE:
CONSITANT1 1.0000E+00
```

Description This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:DEFine

Function Sets or queries an expression for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:DEFine {<String>}
:MATH<x>:USERdefine:DEFine?
```
<x> = 1 to 4
<String> = Up to 128 characters

Example
```
:MATH1:USERDEFINE:DEFINE "C1-C2"
:MATH1:USERDEFINE:DEFINE?
-> :MATH1:USERDEFINE:DEFINE "C1-C2"
```

Description You can only use the characters and symbols on the keyboard that appears on the instrument's screen.

:MATH<x>:USERdefine:FILTer<y>?

Function Queries all settings related to the filter of user-defined computation.

Syntax
```
:MATH<x>:USERdefine:FILTer<y>?
```
<x> = 1 to 4
<y> = 1 or 2

Description This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:FILTer<y>:BAND

Function Sets or queries the filter band (frequency band) for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:FILTer<y>:BA
ND {BPASs|HPASs|LPASs}
:MATH<x>:USERdefine:FILTer<y>:BAND?
```
<x> = 1 to 4
<y> = 1 or 2

Example
```
:MATH1:USERDEFINE:FILTER1:BAND BPASS
:MATH1:USERDEFINE:FILTER1:BAND?
-> :MATH1:USERDEFINE:FILTER1:
BAND BPASS
```

Description This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:FILTer<y>:CUToff<z>

Function Sets or queries an cutoff frequency for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:FILTer<y>:CUToff
<z> {<NRf>}
:MATH<x>:USERdefine:FILTer<y>:CUToff
<z>?
```
<x> = 1 to 4
<y> = 1 or 2
<z> = 1 or 2
<NRf> = 2 to 30 % (in 0.2 steps)

Example
```
:MATH1:USERDEFINE:FILTER1:CUTOFF1 10
:MATH1:USERDEFINE:FILTER1:CUTOFF1?
-> :MATH1:USERDEFINE:FILTER1:CUTO
FF1 10.0
```

Description CUToff2 can only be specified with :MATH<x>:USERdefine:FILTer<x>:BAND BPASs. This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:FILTer<y>:TYPE

Function Sets or queries the filter type for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:FILTer<y>:TY
PE {IIR|FIR}
:MATH<x>:USERdefine:FILTer<y>:TYPE?
```
<x> = 1 to 4
<y> = 1 or 2

Example
```
:MATH1:USERDEFINE:FILTER1:TYPE IIR
:MATH1:USERDEFINE:FILTER1:TYPE?
-> :MATH1:USERDEFINE:FILTER1:
TYPE IIR
```

Description This command applies to MATH1 to MATH4.

:MATH<x>:USERdefine:HISTory:ABORt

Function Aborts the user-defined computation of history waveforms (Math on History).

Syntax
```
:MATH<x>:USERdefine:HISTory:ABORt
```
<x> = 1 to 4

Example
```
:MATH1:USERDEFINE:HISTORY:ABORT
```

:MATH<x>:USERdefine:HISTory:EXECute

Function Executes the user-defined computation of history waveforms (Math on History).

Syntax
```
:MATH<x>:USERdefine:HISTory:EXECute
```
<x> = 1 to 4

Example
```
:MATH1:USERDEFINE:HISTORY:EXECUTE
```

:MATH<x>:USERdefine:SCALe?

Function Queries all settings related to the scaling of user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe?
```
<x> = 1 to 4

:MATH<x>:USERdefine:SCALe:ARANging

Function Executes auto range on user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe:ARANging
```
<x> = 1 to 4

Example
```
:MATH1:USERDEFINE:SCALE:ARANGING
```

:MATH<x>:USERdefine:SCALe:CENTer

Function Sets or queries the center value of scaling for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe:CENT
er {<NRf>}
:MATH<x>:USERdefine:SCALe:CENTer?
```
<x> = 1 to 4
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:MATH1:USERDEFINE:SCALE:CENTER 1
:MATH1:USERDEFINE:SCALE:CENTER?
-> :MATH1:USERDEFINE:SCALE:
CENTER 1.00000E+00
```

:MATH<x>:USERdefine:SCALe:SENSitivity

Function Sets or queries the scaling sensitivity for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe:SENSitivi
ty {<NRf>}
:MATH<x>:USERdefine:SCALe:SENSitivi
ty?
```
<x> = 1 to 4
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:MATH1:USERDEFINE:SCALE:SENSITIVI
TY 10
:MATH1:USERDEFINE:SCALE:SENSITIVITY?
-> :MATH1:USERDEFINE:SCALE:SENSITIVI
TY 10.0000E+00
```

5.20 MEASure Group

The commands dealing with logic waveforms in this group cannot be used on models without logic input.

:MEASure?

Function Queries all the settings for automated measurement of waveform parameters.

Syntax :MEASure?

:MEASure:BIT<x>?

Function Queries all the parameter on/off states of the specified logic waveform.

Syntax :MEASure:BIT<x>?
<x> = 1 to 8

:MEASure:BIT<x>:ALL

Function Collectively turns on or off all measurement items of a logic waveform.

Syntax :MEASure:BIT<x>:ALL {<Boolean>}
<x> = 1 to 8

Example :MEASURE:BIT1:ALL ON

:MEASure:BIT<x>:AREA2?

Function Queries all the parameter on/off states of a logic waveform in Area2.

Syntax :MEASure:BIT<x>:AREA2?
<x> = 1 to 8

:MEASure:BIT<x>:AREA2:ALL

Function Collectively turns on or off all measurement items of a logic waveform in Area2.

Syntax :MEASure:BIT<x>:AREA2:
ALL {<Boolean>}
<x> = 1 to 8

Example :MEASURE:BIT1:AREA2:ALL ON

:MEASure:BIT<x>:AREA2:<Parameter>?

Function Queries a waveform parameter setting of a logic waveform in Area2.

Syntax :MEASure:BIT<x>:AREA2:<Parameter>?
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}

:MEASure:BIT<x>:AREA2:<Parameter>:COUNt?

Function Queries the statistical processing count of a logic waveform parameter in Area2.

Syntax :MEASure:BIT<x>:AREA2:<Parameter>:COUNt?
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}

Example The following is an example of the average frequency of bit 1.
:MEASURE:BIT1:AREA2:AVGFREQ:COUNT?
-> :MEASURE:BIT1:AREA2:AVGFREQ:
COUNT 100

:MEASure:BIT<x>:AREA2:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries a statistical value of a logic waveform parameter in Area2.

Syntax :MEASure:BIT<x>:AREA2:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}

Example The following is an example of the average frequency of bit 1.
:MEASURE:BIT1:AREA2:AVGFREQ:MAXIMUM?
-> :MEASURE:BIT1:AREA2:AVGFREQ:MAXIM
UM 10.0000E+03

Description If the statistical value is immeasurable, the instrument returns “NAN” (not a number).

:MEASure:BIT<x>:AREA2:<Parameter>:STATe

Function Sets or queries the on/of state of a logic waveform parameter in Area2.

Syntax :MEASure:BIT<x>:AREA2:<Parameter>:STATe {<Boolean>}
:MEASure:BIT<x>:AREA2:<Parameter>:STATe?
<x> = 1 to 8
<Parameter> = {AVGFreq|DUTYcycle|FREQuency|PERiod|PNUMber}

Example The following is an example of the average frequency of bit 1.
:MEASURE:BIT1:AREA2:AVGFREQ:STATE ON
:MEASURE:BIT1:AREA2:AVGFREQ:STATE?
-> :MEASURE:BIT1:AREA2:AVGFREQ:
STATE 1

:MEASure:BIT<x>:AREA2:<Parameter>:VALue?

Function Queries automated measurement values of waveform parameters of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:<Parameter>:VALue? [{NRf}]
```
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}
<NRf> = See the Features Guide for this information.

Example The following is an example of the average frequency of bit 1.
```
:MEASURE:BIT1:AREA2:AVGFREQ:VALUE?
-> :MEASURE:BIT1:AREA2:AVGFREQ:VALUE 10.0000E+03
```

Description
- If the value is immeasurable, the instrument returns “NAN” (not a number).
- The <NRf> is used to specify which iteration of automated measurement to query the measured value from.
- If <NRf> is set to 1, the oldest measured value in the automated measurement memory is queried.
- If a measured value does not exist at the specified iteration, the instrument returns “NAN” (not a number).
- If <NRf> is omitted, the most recent measured value is queried.
- If cyclic statistical processing for automated measurement is being executed and <NRf> is specified, the measured values over a cycle in iteration <NRf> from the left of the display on the displayed waveform is queried. If <NRf> is omitted, the measured values over the last cycle on the displayed waveform is queried.

:MEASure:BIT<x>:AREA2:COPY

Function Copies the on/off state of all measurement items of a logic waveform to all other waveforms in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:COPY
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:AREA2:COPY
```

:MEASure:BIT<x>:AREA2:DELay?

Function Queries all settings related to the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay?
```
<x> = 1 to 8

:MEASure:BIT<x>:AREA2:DELay:MEASure?

Function Queries all of the source waveform settings for measuring the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:MEASure?
```
<x> = 1 to 8

:MEASure:BIT<x>:AREA2:DELay:MEASure:COUNt

Function Sets or queries the count number of a source waveform for measuring the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:MEASure:COUNt {<NRf>}
:MEASure:BIT<x>:AREA2:DELay:MEASure:COUNt?
```
<x> = 1 to 8
<NRf> = 1 to 10

Example
```
:MEASURE:BIT1:AREA2:DELAY:MEASURE:COUNT 2
:MEASURE:BIT1:AREA2:DELAY:MEASURE:COUNT?
-> :MEASURE:BIT1:AREA2:DELAY:MEASURE:COUNT 2
```

:MEASure:BIT<x>:AREA2:DELay:MEASure:SLOPe

Function Sets or queries the slope of a source waveform for measuring the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:MEASure:SLOPe {FALL|RISE}
:MEASure:BIT<x>:AREA2:DELay:MEASure:SLOPe?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:AREA2:DELAY:MEASURE:SLOPE FALL
:MEASURE:BIT1:AREA2:DELAY:MEASURE:SLOPE?
-> :MEASURE:BIT1:AREA2:DELAY:MEASURE:SLOPE FALL
```

:MEASure:BIT<x>:AREA2:DELay:REFerence?

Function Queries all of the reference waveform settings for measuring the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:REFerence?
```
<x> = 1 to 8

:MEASure:BIT<x>:AREA2:DELay:REFerence:COUNt

Function Sets or queries the count number of the reference waveform for measuring the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:REFerence:COUNt {<NRf>}
:MEASure:BIT<x>:AREA2:DELay:REFerence:COUNt?
```
<x> = 1 to 8
<NRf> = 1 to 10

Example
```
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:COUNT 2
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:COUNT?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENCE:COUNT 2
```

:MEASure:BIT<x>:AREA2:DELay:REFerence:SLOPe

Function Sets or queries the slope of the reference waveform for measuring the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:REFerence:SLOPe {FALL|RISE}
:MEASure:BIT<x>:AREA2:DELay:REFerence:SLOPe?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:SLOPE FALL
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:SLOPE?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENCE:SLOPE FALL
```

:MEASure:BIT<x>:AREA2:DELay:REFerence:SOURce

Function Sets or queries whether the reference point for measuring the delay between channels of a logic waveform will be set to the trigger point or a waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:REFerence:SOURce {TRACe|TRIGger}
:MEASure:BIT<x>:AREA2:DELay:REFerence:SOURce?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:SOURCE TRACE
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:SOURCE?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENCE:SOURCE TRACE
```

:MEASure:BIT<x>:AREA2:DELay:REFerence:TRACe

Function Sets or queries the edge of the reference waveform for measuring the delay between channels of a logic waveform in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:REFerence:TRACe {<NRf>|BIT<x>|MATH<x>}
:MEASure:BIT<x>:AREA2:DELay:REFerence:TRACe?
```
<x> of BIT<x> = 1 to 8
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:TRACE 2
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:TRACE?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENCE:TRACE 2
```

:MEASure:BIT<x>:AREA2:DELay:STATe

Function Sets or queries the on/off state and display format of the delay parameter in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:STATe {OFF|ON|DEGRee}
:MEASure:BIT<x>:AREA2:DELay:STATe?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:AREA2:DELAY:STATE ON
:MEASURE:BIT1:AREA2:DELAY:STATE?
-> :MEASURE:BIT1:AREA2:DELAY:STATE ON
```

:MEASure:BIT<x>:<Parameter>?

Function Queries a waveform parameter setting of a logic waveform.

Syntax
```
:MEASure:BIT<x>:<Parameter>?
```
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}

:MEASure:BIT<x>:<Parameter>:COUNt?

Function Queries the statistical processing count of a logic waveform parameter.

Syntax
```
:MEASure:BIT<x>:<Parameter>:COUNt?
```
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}

Example The following is an example of the average frequency of bit 1.
```
:MEASURE:BIT1:AVGFREQ:COUNT?
-> :MEASURE:BIT1:AVGFREQ:COUNT 100
```

:MEASure:BIT<x>:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries a statistical value of a logic waveform parameter.

Syntax `:MEASure:BIT<x>:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?`
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}

Example The following is an example of the average frequency of bit 1.
```
:MEASURE:BIT1:AVGFREQ:MAXIMUM?
-> :MEASURE:BIT1:AVGFREQ:
MAXIMUM 10.0000E+03
```

Description If the statistical value cannot be obtained, the instrument returns “NAN” (not a number).

:MEASure:BIT<x>:<Parameter>:STATe

Function Sets or queries the on/of state of a logic waveform parameter.

Syntax `:MEASure:BIT<x>:<Parameter>:STATe {<Boolean>}`
`:MEASure:BIT<x>:<Parameter>:STATe?`
<x> = 1 to 8
<Parameter> = {AVGFreq|DUTYcycle|FREQuency|PERiod|PNUMber}

Example The following is an example of the average frequency of bit 1.
```
:MEASURE:BIT1:AVGFREQ:STATE ON
:MEASURE:BIT1:AVGFREQ:STATE?
-> :MEASURE:BIT1:AVGFREQ:STATE 1
```

:MEASure:BIT<x>:<Parameter>:VALue?

Function Queries automated measurement values of waveform parameters of a logic waveform.

Syntax `:MEASure:BIT<x>:<Parameter>:VALue? [{NRf}]`
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}
<NRf> = See the Features Guide for this information.

Example The following is an example of the average frequency of bit 1.
```
:MEASURE:BIT1:AVGFREQ:VALUE?
-> :MEASURE:BIT1:AVGFREQ:
VALUE 10.0000E+03
```

Description If the value is immeasurable, the instrument returns “NAN” (not a number). The <NRf> at the end is used to specify which iteration of statistical processing to query the parameter value from. This command returns “NAN” if the specified value does not exist.

- For measurements other than cyclic statistical processing <NRf> can be omitted. If you omit , the most recent history parameter value will be queried. If <NRf> is included, the parameter value of the waveform <NRf> times before the newest waveform will be queried from the acquisition memory.
- For measurements after cyclic statistical processing <NRf> can be omitted. If you omit , the parameter value within the cycle that was measured last will be queried. If you include <NRf>, the parameter value within the cycle that was measured <NRf> times after the measurement shown on the screen's left edge will be queried.

:MEASure:BIT<x>:COPY

Function Copies the on/off state of all measurement items of a logic waveform to all other waveforms.

Syntax `:MEASure:BIT<x>:COPY`
<x> = 1 to 8

Example `:MEASURE:BIT1:COPY`

:MEASure:BIT<x>:DELay?

Function Queries all settings related to the delay between channels of a logic waveform.

Syntax `:MEASure:BIT<x>:DELay?`
<x> = 1 to 8

:MEASure:BIT<x>:DELay:MEASure?

Function Queries all the settings for a source waveform for measuring the delay between channels of a logic waveform.

Syntax `:MEASure:BIT<x>:DELay:MEASure?`
<x> = 1 to 8

:MEASure:BIT<x>:DELay:MEASure:COUNt

Function Sets or queries the source waveform count for measuring the delay between channels of a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:MEASure:COU
Nt {<NRf>}
:MEASure:BIT<x>:DELay:MEASure:COUNt?
```
<x> = 1 to 8
<NRf> = 1 to 10

Example
```
:MEASURE:BIT1:DELAY:MEASURE:COUNT 2
:MEASURE:BIT1:DELAY:MEASURE:COUNT?
-> :MEASURE:BIT1:DELAY:MEASURE:
COUNT 2
```

:MEASure:BIT<x>:DELay:MEASure:SLOPe

Function Sets or queries a source waveform slope that will be used to measure delay between channels for a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:MEASure:SLO
Pe {FALL|RISE}
:MEASure:BIT<x>:DELay:MEASure:SLOPe?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:DELAY:MEASURE:
SLOPE FALL
:MEASURE:BIT1:DELAY:MEASURE:SLOPE?
-> :MEASURE:BIT1:DELAY:MEASURE:
SLOPE FALL
```

:MEASure:BIT<x>:DELay:REFerence?

Function Queries all the settings for the reference waveform for measuring the delay between channels of a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence?
```
<x> = 1 to 8

:MEASure:BIT<x>:DELay:REFerence:COUNt

Function Sets or queries the reference waveform count for measuring the delay between channels of a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:COU
Nt {<NRf>}
:MEASure:BIT<x>:DELay:REFerence:COU
Nt?
```
<x> = 1 to 8
<NRf> = 1 to 10

Example
```
:MEASURE:BIT1:DELAY:REFERENCE:COU
NT 2
:MEASURE:BIT1:DELAY:REFERENCE:COUNT?
-> :MEASURE:BIT1:DELAY:REFERENCE:COU
NT 2
```

:MEASure:BIT<x>:DELay:REFerence:SLOPe

Function Sets or queries the reference waveform slope that will be used to measure delay between channels for a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:SLO
Pe {FALL|RISE}
:MEASure:BIT<x>:DELay:REFerence:SLO
Pe?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:DELAY:REFERENCE:SLO
PE FALL
:MEASURE:BIT1:DELAY:REFERENCE:SLOPE?
-> :MEASURE:BIT1:DELAY:REFERENCE:SLO
PE FALL
```

:MEASure:BIT<x>:DELay:REFerence:SOURce

Function Sets or queries whether the reference point for measuring the delay between channels of a logic waveform will be set to the trigger point or a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:SOUR
ce {TRACe|TRIGger}
:MEASure:BIT<x>:DELay:REFerence:SOUR
ce?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:DELAY:REFERENCE:SOUR
CE TRACE
:MEASURE:BIT1:DELAY:REFERENCE:SOUR
CE?
-> :MEASURE:BIT1:DELAY:REFERENCE:SOU
RCE TRACE
```

:MEASure:BIT<x>:DELay:REFerence:TRACe

Function Sets or queries the reference waveform edge that will be used to measure delay between channels for a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:TRA
Ce {<NRf>|BIT<y>|MATH<y>}
:MEASure:BIT<x>:DELay:REFerence:TRA
Ce?
```
<x> = 1 to 8
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:BIT1:DELAY:REFERENCE:TRA
CE 2
:MEASURE:BIT1:DELAY:REFERENCE:TRACE?
-> :MEASURE:BIT1:DELAY:REFERENCE:TRA
CE 2
```

:MEASure:BIT<x>:DELay:STATe

Function Sets or queries the on/off state and display format of the delay parameter.

Syntax `:MEASure:BIT<x>:DELay:STATe {OFF|ON|DEGRee}`
`:MEASure:BIT<x>:DELay:STATe?`
<x> = 1 to 8

Example
```
:MEASURE:BIT1:DELAY:STATE ON
:MEASURE:BIT1:DELAY:STATE?
-> :MEASURE:BIT1:DELAY:STATE ON
```

:MEASure:{CHANnel<x>|MATH<x>}?

Function Queries all the parameter on/off states of a waveform.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:ALL

Function Collectively turns on or off all measurement items of a waveform.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:ALL {<Boolean>}`
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:ALL ON
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2?

Function Queries all the parameter on/off states of a waveform in Area2.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:AREA2?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:ALL

Function Collectively turns on or off all measurement items of a waveform in Area2.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:AREA2:ALL {<Boolean>}`
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:ALL ON
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>?

Function Queries a waveform parameter setting of a waveform in Area2.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:COUNt?

Function Queries the statistical processing count of a waveform parameter in Area2.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:COUNt?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AREA2:AVERAGE:COUNT?
-> :MEASURE:CHANNEL1:AREA2:AVERAGE:COUNT 100
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries a statistical value of a waveform parameter in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AREA2:AVERAGE:MAXIMUM?
-> :MEASURE:CHANNEL1:AREA2:AVERAGE:MAXIMUM 10.0000E+03
```

Description If the statistical value cannot be obtained, the instrument returns “NAN” (not a number).

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:STATe

Function Sets or queries the on/of state of a waveform parameter in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:STATe {<Boolean>}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:STATe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AREA2:AVERAGE:STATE ON
:MEASURE:CHANNEL1:AREA2:AVERAGE:STATE?
-> :MEASURE:CHANNEL1:AREA2:AVERAGE:STATE 1
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:VALue?

Function Queries the automated measurement value of a waveform parameter in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:VALue? [<NRf>[,STATus]]
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:VALue? [STATus]
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}
<NRf> = See the Features Guide for this information.

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AREA2:AVERAGE:VALUE?
-> :MEASURE:CHANNEL1:AREA2:AVERAGE:VALUE 10.0000E+03
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:COPY

Function Copies the on/off state of all measurement items of a waveform to all other waveforms in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:COPY
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:COPY
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:CYCLe

Function Sets or queries cycle mode of Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:CYCLe {ONE|N|OFF}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:CYCLe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:CYCLE ONE
:MEASURE:CHANNEL1:AREA2:CYCLE?
-> :MEASURE:CHANNEL1:AREA2:CYCLE ONE
```

Description
- When :ANALysis:PANalyze:TYPE SWLoss is specified, “SWLOSS” is returned in response to a query.
- If :ANALysis:PMEAsure<x>:MODE ON or :ANALysis:PMEAsure:CYCMode NCYCle is specified, “NCCHANNEL” is returned in response to a query.

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay?

Function Queries all settings related to the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure?

Function Queries all of the source waveform settings for measuring the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:COUNt

Function Sets or queries the count number of a source waveform for measuring the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:COUNt {<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:COUNt?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1 to 10

Example
```
:MEASURE:CHANNEL1:AREA2:DELAY:MEASURE:COUNT 2
:MEASURE:CHANNEL1:AREA2:DELAY:MEASURE:COUNT?
-> :MEASURE:CHANNEL1:AREA2:DELAY:MEASURE:COUNT 2
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:SLOPe

Function Sets or queries the slope of a source waveform for measuring the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:SLOPe {FALL|RISE}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:SLOPe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:DELAY:MEASURE:SLOPE FALL
:MEASURE:CHANNEL1:AREA2:DELAY:MEASURE:SLOPE?
-> :MEASURE:CHANNEL1:AREA2:DELAY:MEASURE:SLOPE FALL
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence?

Function Queries all of the reference waveform settings for measuring the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:COUNt

Function Sets or queries the count number of the reference waveform for measuring the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:COUNt {<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:COUNt?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1 to 10

Example
```
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:COUNT 2
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:COUNT?
-> :MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:COUNT 2
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SLOPe

Function Sets or queries the slope of the reference waveform for measuring the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SLOPe {FALL|RISE}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SLOPe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:SLOPE FALL
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:SLOPE?
-> :MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:SLOPE FALL
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SOURce

Function Sets or queries whether the reference point for measuring the delay between channels of a waveform will be set to the trigger point or a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SOURce {TRACe|TRIGger}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SOURce?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:SOURCE TRACE
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:SOURCE?
-> :MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:SOURCE TRACE
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:TRACe

Function Sets or queries the edge of the reference waveform for measuring the delay between channels of a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:TRACe {<NRf>|MATH<x>}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:TRACe?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:TRACE 2
:MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:TRACE?
-> :MEASURE:CHANNEL1:AREA2:DELAY:REFERENCE:TRACE 2
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:STATe

Function Sets or queries the on/off state and display format of the delay parameter in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:STATe {OFF|ON|DEGRee}
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:STATe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:DELAY:STATE ON
:MEASURE:CHANNEL1:AREA2:DELAY:STATE?
-> :MEASURE:CHANNEL1:AREA2:DELAY:STATE ON
```

:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>?

Function Queries a waveform parameter setting of a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:COUNt?

Function Queries the statistical processing count of a waveform parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:COUNt?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AVERAGE:COUNT?
-> :MEASURE:CHANNEL1:AVERAGE:COUNT 100
```

:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries a statistical value of a waveform parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AVERAGE:MAXIMUM?
-> :MEASURE:CHANNEL1:AVERAGE:MAXIMUM 10.0000E+03
```

Description If the statistical value cannot be obtained, the instrument returns “NAN” (not a number).

:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:STATe

Function Sets or queries the on/of state of a waveform parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:STATe {<Boolean>}
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:STATe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AVERAGE:STATE ON
:MEASURE:CHANNEL1:AVERAGE:STATE?
-> :MEASURE:CHANNEL1:AVERAGE:STATE 1
```

:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:VALue?

Function Queries automated measurement values of waveform parameters of a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:VALue? [<NRf>[,STATus]]
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:VALue? [STATus]
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}
<NRf> = See the Features Guide for this information.

Example The following is an example of the average value of CH1.
```
:MEASURE:CHANNEL1:AVERAGE:VALUE?
-> :MEASURE:CHANNEL1:AVERAGE:
VALUE 10.0000E+03
```

Description
- If the value is immeasurable, the instrument returns “NAN” (not a number).
- The <NRf> is used to specify which iteration of automated measurement to query the measured value from.
- If <NRf> is set to 1, the oldest measured value in the automated measurement memory is queried.
- If a measured value does not exist at the specified iteration, the instrument returns “NAN” (not a number).
- If <NRf> is omitted, the most recent measured value is queried.
- If cyclic statistical processing for automated measurement is being executed and <NRf> is specified, the measured values over a cycle in iteration <NRf> from the left of the display on the displayed waveform is queried. If <NRf> is omitted, the measured values over the last cycle on the displayed waveform is queried.
- The optional “STATus” at the end is valid when the parameter is set to FALL, NWIDth, PERiod, PWIDth, or RISE. If “STATus” is omitted for these parameters, if the measurement value is less than the measurement resolution, a negative value is returned. If “STATus” in included, “LOW_RESOL” is returned.

:MEASure:{CHANnel<x>|MATH<x>}:COPY

Function Copies the on/off state of all measurement items of a waveform to all other waveforms.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:COPY
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:COPY
```

:MEASure:{CHANnel<x>|MATH<x>}:CYCLe

Function Sets or queries the cycle mode.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:CYCLe {ONE|N|OFF}
:MEASure:{CHANnel<x>|MATH<x>}:CYCLe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:CYCLE ONE
:MEASURE:CHANNEL1:CYCLE?
-> :MEASURE:CHANNEL1:CYCLE ONE
```

Description
- When :ANALysis:PANalyze:TYPE SWLoss is specified, “SWLOSS” is returned in response to a query.
- If :ANALysis:PMEAsure<x>:MODE ON and :ANALysis:PMEAsure:CYCMode NCYCle are specified, “NCCHANNEL” is returned in response to a query.

:MEASure:{CHANnel<x>|MATH<x>}:DELay?

Function Queries all settings related to the delay between channels of a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure?

Function Queries all the settings for a source waveform for measuring the delay between channels of a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:COUNt

Function Sets or queries the number of edges at which delay between channels will be measured for a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:COUNt {<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:COUNt?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1 to 10

Example
```
:MEASURE:CHANNEL1:DELAY:MEASURE:COUNT 2
:MEASURE:CHANNEL1:DELAY:MEASURE:COUNT?
-> :MEASURE:CHANNEL1:DELAY:MEASURE:COUNT 2
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:SLOPe

Function Sets or queries a source waveform slope that will be used to measure delay between channels for a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:SLOPe {FALL|RISE}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:SLOPe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:MEASURE:SLOPE FALL
:MEASURE:CHANNEL1:DELAY:MEASURE:SLOPE?
-> :MEASURE:CHANNEL1:DELAY:MEASURE:SLOPE FALL
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence?

Function Queries all reference waveform settings used to measure the delay between channels for the specified waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:COUNt

Function Sets or queries the reference waveform count for measuring the delay between channels of a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:COUNt {<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:COUNt?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1 to 10

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:COUNT 2
:MEASURE:CHANNEL1:DELAY:REFERENCE:COUNT?
-> :MEASURE:CHANNEL1:DELAY:REFERENCE:COUNT 2
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SLOPe

Function Sets or queries the reference waveform slope that will be used to measure delay between channels for a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SLOPe {FALL|RISE}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SLOPe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:SLOPE FALL
:MEASURE:CHANNEL1:DELAY:REFERENCE:SLOPE?
-> :MEASURE:CHANNEL1:DELAY:REFERENCE:SLOPE FALL
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SOURce

Function Sets or queries whether to set the reference point for measuring the delay between channels for a waveform to a trigger point or to a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SOURce {TRACe|TRIGger}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SOURce?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:SOURCE TRACE
:MEASURE:CHANNEL1:DELAY:REFERENCE:SOURCE?
-> :MEASURE:CHANNEL1:DELAY:REFERENCE:SOURCE TRACE
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:TRACe

Function Sets or queries the reference waveform edge that will be used to measure delay between channels for a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:TRACe {<NRf>|MATH<x>}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:TRACe?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:TRACE 2
:MEASURE:CHANNEL1:DELAY:REFERENCE:TRACE?
-> :MEASURE:CHANNEL1:DELAY:REFERENCE:TRACE 2
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:STATe

Function Sets or queries the on/off state and display format of the delay parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:STATe {OFF|ON|DEGRee}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:STATe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:STATE ON
:MEASURE:CHANNEL1:DELAY:STATE?
-> :MEASURE:CHANNEL1:DELAY:STATE ON
```

:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal?

Function Queries all distal, mesial, and proximal settings.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:MODE

Function Sets or queries the distal, mesial, and proximal point mode setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:MODE {PERCent|UNIT}
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:MODE?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:DPROXIMAL:MODE PERCENT
:MEASURE:CHANNEL1:DPROXIMAL:MODE?
-> :MEASURE:CHANNEL1:DPROXIMAL:MODE PERCENT
```

:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:PERCent

Function Sets or queries the distal, mesial, and proximal points as percentages.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:PERCent {<NRf>,<NRf>,<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:PERCent?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 0 to 100 (%, in steps of 1)

Example
```
:MEASURE:CHANNEL1:DPROXIMAL:PERCENT 80,60,40
:MEASURE:CHANNEL1:DPROXIMAL:PERCENT?
-> :MEASURE:CHANNEL1:DPROXIMAL:PERCENT 40,60,80
```

:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:UNIT

Function Sets or queries the distal, mesial, and proximal points as voltages.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:UNIT {<Voltage>,<Voltage>,<Voltage>|<NRf>,<NRf>,<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:UNIT?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Voltage>, <NRf> = See the Features Guide for this information.

Example
```
:MEASURE:CHANNEL1:DPROXIMAL:UNIT 50V,0V,-50V
:MEASURE:CHANNEL1:DPROXIMAL:UNIT?
-> :MEASURE:CHANNEL1:DPROXIMAL:UNIT -50.0E+00,0.0E+00,50.0E+00
```

Description When a current probe is in use, this command does not set or query the <Current> value.

:MEASure:{CHANnel<x>|MATH<x>}:METHod

Function Sets or queries the high and low points.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:METHod {AUTO|MAXMin|HISTogram}
:MEASure:{CHANnel<x>|MATH<x>}:METHod?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CHANNEL1:METHOD AUTO
:MEASURE:CHANNEL1:METHOD?
-> :MEASURE:CHANNEL1:METHOD AUTO
```

:MEASure:CONTinuous?

Function Queries all settings for the normal statistical processing of automatically measured waveform parameters.

Syntax
```
:MEASure:CONTinuous?
```

:MEASure:CONTinuous:RESTart

Function Restarts the normal statistical processing of automatically measured waveform parameters.

Syntax
```
:MEASure:CONTinuous:RESTart
:MEASURE:CONTINUOUS:RESTART
```

:MEASure:CONTinuous:TLCHange (TriggerLevel Change)

Function Sets or queries whether normal statistical processing of automatically measured waveform parameters is restarted when the trigger level is changed.

Syntax
```
:MEASure:CONTinuous:TLCHange{RESTart|IGNore}:MEASure:CONTinuous:TLCHange?
```

Example
```
:MEASURE:CONTINUOUS:TLCHANGE RESTART
:MEASURE:CONTINUOUS:TLCHANGE?
-> :MEASURE:CONTINUOUS:TLCHANGE RESTART
```

:MEASure:CYCLe?

Function Queries all settings related to the cyclic statistical processing of automated measurement.

Syntax
```
:MEASure:CYCLe?
```

:MEASure:CYCLe:ABORt

Function Aborts the execution of the cyclic statistical processing of automated measurement.

Syntax
```
:MEASure:CYCLe:ABORt
```

Example
```
:MEASURE:CYCLE:ABORT
```

:MEASure:CYCLe:EXECute

Function Executes the cyclic statistical processing of automated measurement.

Syntax
```
:MEASure:CYCLe:EXECute
```

Example
```
:MEASURE:CYCLE:EXECUTE
```

:MEASure:CYCLe:TRACe

Function Sets or queries the source waveform of the cyclic statistical processing of automated measurement.

Syntax
```
:MEASure:CYCLe:TRACe {OWN|<NRf>|MATH<x>}
:MEASure:CYCLe:TRACe?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:MEASURE:CYCLE:TRACE 1
:MEASURE:CYCLE:TRACE?
-> :MEASURE:CYCLE:TRACE 1
```

:MEASure:HISTory:ABORt

Function Aborts the execution of the statistical processing of the history waveform of automated measurement.

Syntax
```
:MEASure:HISTory:ABORt
:MEASURE:HISTORY:ABORT
```

:MEASure:HISTory:EXECute

Function Executes the statistical processing of the history waveform of automated measurement.

Syntax
```
:MEASure:HISTory:EXECute
```

Example
```
:MEASURE:HISTORY:EXECUTE
```

:MEASure:INDicator

Function Sets or queries the measurement location indicator.

Syntax
```
:MEASure:INDicator {<NRf>|BIT<x>|MATH<x>|OFF{,<Parameter>,2}}
:MEASure:INDicator?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DUTYcycle|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|POVershoot|PTOPeak|PWIDth|RISE|RMS|V1|V2}

Example
```
:MEASURE:INDICATOR 1,AVERAGE
:MEASURE:INDICATOR?
-> :MEASURE:INDICATOR 1,AVERAGE
```

:MEASure:MODE

Function Sets or queries the automated measurement mode.

Syntax
```
:MEASure:MODE {OFF|ON|CONTinuous|CYCLe|HISTory}
:MEASure:MODE?
```

Example
```
:MEASURE:MODE OFF
:MEASURE:MODE?
-> :MEASURE:MODE OFF
```

:MEASure:RANGe<x>

Function Sets or queries the measurement source window.

Syntax
```
:MEASure:RANGe<x> {MAIN|Z1|Z2}
:MEASure:RANGe<x>?
```
<x> = 1 or 2

Example
```
:MEASURE:RANGE1 MAIN
:MEASURE:RANGE1?
-> :MEASURE:RANGE1 MAIN
```

Description ":Measure:RANGe1" is for setting or querying the measurement source window of Area1 for normal measurement and two area measurement. ":Measure:RANGe2" is for setting or querying the measurement source window of Area2 for two area measurement.

:MEASure:TRANge<x> (Time Range)

Function Sets or queries the measurement time period.

Syntax
```
:MEASure:TRANge<x> {<NRf>,<NRf>}
:MEASure:TRANge<x>?
```
<x> = 1 or 2
<NRf> = −5 div to 5 div (in steps of 10 divisions/display record length)

Example
```
:MEASURE:TRANGE1 -4,4
:MEASURE:TRANGE1?
-> :MEASURE:TRANGE1 -4.00E+00,4.00E+00
```

Description ":Measure:TRANge1" is for setting or querying the measurement range of Area1 for normal measurement and two area measurement. ":Measure:TRANge2" is for setting or querying the measurement range of Area2 for two area measurement.

:MEASure:USER<x>?

Function Queries all automatic measurement settings for a Calc item.

Syntax
```
:MEASure:USER<x>?
```
<x> = 1 to 4

:MEASure:USER<x>:COUNt?

Function Queries the statistical processing count of the automatically measured value of a Calc item.

Syntax
```
:MEASure:USER<x>:COUNt?
```
<x> = 1 to 4

Example
```
:MEASURE:USER1:COUNT?
-> :MEASURE:USER1:COUNT 100
```

:MEASure:USER<x>:DEFine

Function Sets or queries the expression for the automatically measured value of a Calc item.

Syntax
```
:MEASure:USER<x>:DEFine {<String>}
:MEASure:USER<x>:DEFine?
```
<x> = 1 to 4
<String> = Up to 128 characters

Example
```
:MEASURE:USER1:DEFINE "MAX(C1)"
:MEASURE:USER1:DEFINE?
-> :MEASURE:USER1:DEFINE "MAX(C1)"
```

:MEASure:USER<x>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries a statistical value that is calculated on the automatically measured value of a Calc item.

Syntax
```
:MEASure:USER<x>:MAXimum?
```
<x> = 1 to 4

Example The following example is for the maximum value.
```
:MEASURE:USER1:MAXIMUM?
-> :MEASURE:USER1:MAXIMUM 1.000E+00
```

Description If the statistical value cannot be obtained, the instrument returns "NAN" (not a number).

:MEASure:USER<X>:NAME

Function Sets or queries the name of a Calc item.

Syntax
```
:MEASure:USER<X>:NAME {<String>}
:MEASure:USER<X>:NAME?
```
<x> = 1 to 4
<String> = Up to 8 characters

Example
```
:MEASURE:USER1:NAME "MAX"
:MEASURE:USER1:NAME?
-> :MEASURE:USER1:NAME "MAX"
```

:MEASure:USER<x>:STATe

Function Sets or queries the on/off status of automated measurement of a Calc item.

Syntax
```
:MEASure:USER<x>:STATe {<Boolean>}
:MEASure:USER<x>:STATe?
```
<x> = 1 to 4

Example
```
:MEASURE:USER1:STATE ON
:MEASURE:USER1:STATE?
-> :MEASURE:USER1:STATE 1
```

:MEASure:USER<x>:UNIT

Function Sets or queries the unit of a Calc item.

Syntax
```
:MEASure:USER<x>:UNIT {<String>}
:MEASure:USER<x>:UNIT?
```
<x> = 1 to 4
<String> = Up to 4 characters

Example
```
:MEASURE:USER1:UNIT "V"
:MEASURE:USER1:UNIT?
-> :MEASURE:USER1:UNIT "V"
```

Description Measured values are never affected by the unit.

:MEASure:USER<x>:VALue?

Function Queries the automatically measured value of a Calc item.

Syntax
```
:MEASure:USER<x>:VALue? [<NRf>]
```
<x> = 1 to 4
<NRf> = See the Features Guide for this information.

Example
```
:MEASURE:USER1:VALUE?
-> :MEASURE:USER1:VALUE 1.000E+00
```

Description
- If the value is immeasurable, the instrument returns “NAN” (not a number).
- The <NRf> is used to specify which iteration of automated measurement to query the measured value from.
- If <NRf> is set to 1, the oldest measured value in the automated measurement memory is queried.
- If a measured value does not exist at the specified iteration, the instrument returns “NAN” (not a number).
- If <NRf> is omitted, the most recent measured value is queried.

:MEASure:WAIT?

Function Waits for the completion of automated measurement with a timeout.

Syntax
```
:MEASure:WAIT? {<NRf>}
```
<NRf> = 1 to 36000 (timeout value, in units of 100 ms)

Example
```
:MEASURE:WAIT?
-> :MEASURE:WAIT 1
```

Description
- If the execution of automated measurement is completed within the timeout period, 0 is returned. If it is not completed or if automated measurement is not being performed, 1 is returned.
- Even if you set a long timeout 0 is returned as soon as the automated measurement is completed.

:MEASure:ZLINkage

Function Sets or queries the linkage (on/off) between the measurement numbers and zoom positions of the results of automated measurement of waveform parameters.

Syntax
```
:MEASure:ZLINkage {<Boolean>}
:MEASure:ZLINkage?
```

Example
```
:MEASURE:ZLINKAGE ON
:MEASURE:ZLINKAGE?
-> :MEASURE:ZLINKAGE 1
```

5.21 RECall Group

:RECall:SETup<x>:EXECute

Function Recalls setup data from the internal memory.

Syntax `:RECall:SETup<x>:EXECute`

<x> = 1 to 3

Example `:RECALL:SETUP1:EXECUTE`

Description If you specify an area that does not contain setup data, an error occurs.

5.22 REFerence Group

:REFerence<x>?

Function Queries all reference waveform settings.

Syntax `:REFerence<x>?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

:REFerence<x>:DISPlay

Function Sets or queries the reference waveform display on/off state.

Syntax `:REFerence<x>:DISPlay {<Boolean>}`
`:REFerence<x>:DISPlay?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:REFERENCE1:DISPLAY ON
:REFERENCE1:DISPLAY?
-> :REFERENCE1:DISPLAY 1
```

:REFerence<x>:LABel?

Function Queries all settings related to the label of a reference.

Syntax `:REFerence<x>:LABel`
<x> = 1 to 4 (1 or 2 on 2-channel models)

:REFerence<x>:LABel[:DEFine]

Function Sets or queries the label of a reference.

Syntax `:REFerence<x>:LABel[:DEFine] {<String>}`
`:REFerence<x>:LABel[:DEFine]?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<String> = Up to 8 characters

Example
```
:REFERENCE1:LABEL:DEFINE "REF1"
:REFERENCE1:LABEL:DEFINE?
-> :REFERENCE1:LABEL:DEFINE "REF1"
```

:REFerence<x>:LABel:MODE

Function Sets or queries the label display on/off state of a reference.

Syntax `:REFerence<x>:LABel:MODE {<Boolean>}`
`:REFerence<x>:LABel:MODE?`
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:REFERENCE1:LABEL:MODE ON
:REFERENCE1:LABEL:MODE?
-> :REFERENCE1:LABEL:MODE 1
```

:REFerence<x>:LOAD

Function Loads a reference waveform.

Syntax `:REFerence<x>:LOAD {<NRf>|MATH<y>}`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> = 1 to 4 (1 or 2 on 2-channel models)

Example `:REFERENCE1:LOAD 1`

Description
- When <X> is 1, <y> can be set between 2 and 4.
- When <X> is 2, <y> can be set 1, 3, or 4.
- When <X> is 3, <y> can be set 1, 2, or 4.
- When <X> is 4, <y> can be set between 1 and 3.

:REFerence<x>:POSition

Function Sets or queries the vertical position of the reference waveform.

Syntax `:REFerence<x>:POSition {<NRf>}`
`:REFerence<x>:POSition?`
<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = –4 div to 4 div

Example
```
:REFERENCE1:POSITION 1
:REFERENCE1:POSITION?
-> :REFERENCE1:POSITION 1.000E+00
```

5.23 SEARch Group

The following table indicates how the communication commands for search methods correspond to the search types that appear when you press a panel key.

Search Type	Communication Command	Setup Menu	
		Panel Key	Search Type
Edge	EDGE	SEARCH Key	Edge
Combination of multiple inputs (pattern)	PATTern		Pattern
Pulse width	PULSe		Pulse Width
Timeout period	TIMeout		Timeout

:SEARch?
Function Queries all waveform search settings.
Syntax `:SEARch?`

:SEARch:ABORt
Function Aborts the search.
Syntax `:SEARch:ABORt`
Example `:SEARCH:ABORT`

:SEARch:ASCRoll<x>?
Function Queries all auto scroll settings.
Syntax `:SEARch:ASCRoll<x>?`
<x> = 1 or 2

:SEARch:ASCRoll<x>:JUMP
Function Moves the center position of the zoom box to the left or right edge of the main window.
Syntax `:SEARch:ASCRoll<x>:JUMP {LEFT|RIGHt}`
<x> = 1 or 2
Example `:SEARCH:ASCROLL1:JUMP LEFT`

:SEARch:ASCRoll<x>:SPEed
Function Sets or queries the auto scroll speed of the zoom box.
Syntax `:SEARch:ASCRoll<x>:SPEed {<NRf>}`
`:SEARch:ASCRoll<x>:SPEed?`
<x> = 1 or 2
<NRf> = 1 to 50
Example `:SEARCH:ASCROLL1:SPEED 1`
`:SEARCH:ASCROLL1:SPEED?`
`-> :SEARCH:ASCROLL1:SPEED 1`

:SEARch:ASCRoll<x>:STARt
Function Starts auto scrolling.
Syntax `:SEARch:ASCRoll<x>:STARt {LEFT|RIGHt}`
<x> = 1 or 2
Example `:SEARCH:ASCROLL1:START LEFT`

:SEARch:ASCRoll<x>:STOP
Function Stops auto scrolling.
Syntax `:SEARch:ASCRoll<x>:STOP`
<x> = 1 or 2
Example `:SEARCH:ASCROLL1:STOP`

:SEARch:EDGE?
Function Queries all edge search settings.
Syntax `:SEARch:EDGE?`

:SEARch:EDGE:HYSTeresis
Function Sets or queries the edge search level hysteresis.
Syntax `:SEARch:EDGE:HYSTeresis {<NRf>}`
`:SEARch:EDGE:HYSTeresis?`
<NRf> = 0.3 to 8.0 div (in 0.1 steps)
Example `:SEARCH:EDGE:HYSTERESIS 1`
`:SEARCH:EDGE:HYSTERESIS?`
`-> :SEARCH:EDGE:HYSTERESIS 1.0`

:SEARch:EDGE:LEVel
Function Sets or queries the edge search level.
Syntax `:SEARch:EDGE:LEVel {<NRf>|<Voltage>}`
`:SEARch:EDGE:LEVel?`
<NRf> = See the Features Guide for this information.
Example `:SEARCH:EDGE:LEVEL -10mV`
`:SEARCH:EDGE:LEVEL?`
`-> :SEARCH:EDGE:LEVEL -10.00E-03`
Description When a current probe is in use, this command does not set or query the <Current> value.

:SEARch:EDGE:SLOPe
Function Sets or queries the edge search slope.
Syntax `:SEARch:EDGE:SLOPe {BOTH|FALL|RISE}`
`:SEARch:EDGE:SLOPe?`
Example `:SEARCH:EDGE:SLOPE RISE`
`:SEARCH:EDGE:SLOPE?`
`-> :SEARCH:EDGE:SLOPE RISE`

:SEARch:EDGE:SOURce

Function Sets or queries the source waveform of edge search.

Syntax
```
:SEARch:EDGE:SOURce {<NRf>|BIT<x>|
MATH<x>}
:SEARch:EDGE:SOURce?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:SEARCH:EDGE:SOURCE 1
:SEARCH:EDGE:SOURCE?
-> :SEARCH:EDGE:SOURCE 1
```

:SEARch:EPOint

Function Sets or queries the search end position.

Syntax
```
:SEARch:EPOint {<NRf>}
:SEARch:EPOint?
```
<NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:SEARCH:EPOINT 0
:SEARCH:EPOINT?
-> :SEARCH:EPOINT 0.000E+00
```

:SEARch:EXECute

Function Executes a search. This is an overlap command.

Syntax
```
:SEARch:EXECute
```

Example
```
:SEARCH:EXECUTE
```

:SEARch:MARK

Function Sets or queries the detected point mark on/off state.

Syntax
```
:SEARch:MARK {<Boolean>}
:SEARch:MARK?
```

Example
```
:SEARCH:MARK ON
:SEARCH:MARK?
-> :SEARCH:MARK 1
```

:SEARch:MAG<x>

Function Sets or queries the zoom factor of the zoom window.

Syntax
```
:SEARch:MAG<x> {<NRf>}
:SEARch:MAG<x>?
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SEARCH:MAG1 2
:SEARCH:MAG1?
-> :SEARCH:MAG1 2
```

:SEARch:POSition<x>

Function Sets or queries the position of a zoom box.

Syntax
```
:SEARch:POSition<x> {<NRf>}
:SEARch:POSition<x>?
```
<x> = 1 or 2
<NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:SEARCH:POSITION1 1
:SEARCH:POSITION1?
-> :SEARCH:POSITION1 1.000E+00
```

:SEARch:PATTern?

Function Queries all settings related to the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern?
```

:SEARch:PATTern:BITS?

Function Queries all settings related to the bits of the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:BITS?
```

:SEARch:PATTern:BITS:HEXa

Function Sets the bit pattern for the combination (pattern) search of multiple inputs in hexadecimal notation.

Syntax
```
:SEARch:PATTern:BITS:HEXa {<String>}
```

Example
```
:SEARCH:PATTERN:BITS:HEXA "AB"
```

:SEARch:PATTern:BITS:PATTern

Function Sets or queries the bit pattern for the combination (pattern) search of multiple inputs in binary notation.

Syntax
```
:SEARch:PATTern:BITS:PATTern {<String>
}
:SEARch:PATTern:BITS:PATTern?
```

Example
```
:SEARCH:PATTERN:BITS:
PATTERN "1100xx01"
:SEARCH:PATTERN:BITS:PATTERN?
-> :SEARCH:PATTERN:BITS:
PATTERN "1100xx01"
```

:SEARch:PATTern:{CHANnel<x>|MATH<x>}?

Function Queries all settings related to a waveform of the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:{CHANnel<x>|MATH
<x>}?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

:SEARch:PATTern:{CHANnel<x>|MATH<x>}:HYSTeresis

Function Sets or queries the hysteresis of the source waveform for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:{CHANnel<x>|MATH<x>}
:HYSTeresis {<NRf>}
:SEARch:PATTern:{CHANnel<x>|MATH<x>}
:HYSTeresis?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = 0.3 to 8.0 div (in 0.1 steps)

Example
```
:SEARCH:PATTERN:CHANNEL1:HYSTERES
IS 0.3
:SEARCH:PATTERN:CHANNEL1:HYSTERESIS?
-> :SEARCH:PATTERN:CHANNEL1:HYSTERES
IS 0.30
```

:SEARch:PATTern:{CHANnel<x>|MATH<x>}:LEVel

Function Sets or queries the search level of the source waveform for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:{CHANnel<x>|MATH<x>}
:LEVel {<NRf>|<Voltage>}
:SEARch:PATTern:{CHANnel<x>|MATH<x>}
:LEVel?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = See the Features Guide for this information.

Example
```
:SEARCH:PATTERN:CHANNEL1:LEVEL 1V
:SEARCH:PATTERN:CHANNEL1:LEVEL?
-> :SEARCH:PATTERN:CHANNEL1:
LEVEL 1.000E+00
```

Description When a current probe is in use, this command does not set or query the <Current> value.

:SEARch:PATTern:{CHANnel<x>|MATH<x>}:PATTern

Function Sets or queries the search pattern of the source waveform for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:{CHANnel<x>|MATH<x>}
:PATTern {DONTcare|HIGH|LOW}
:SEARch:PATTern:{CHANnel<x>|MATH<x>}
:PATTern?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 on 2-channel models)
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:SEARCH:PATTERN:CHANNEL1:
PATTERN HIGH
:SEARCH:PATTERN:CHANNEL1:PATTERN?
-> :SEARCH:PATTERN:CHANNEL1:
PATTERN HIGH
```

:SEARch:PATTern:CLOCk?

Function Queries all settings related to the clock channel of the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:CLOCk?
```

:SEARch:PATTern:CLOCk:HYSTeresis

Function Sets or queries the hysteresis of the clock channel for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:CLOCk:
HYSTeresis {<NRf>}
:SEARch:PATTern:CLOCk:HYSTeresis?
```
<NRf> = 0.3 to 8.0 div (in 0.1 steps)

Example
```
:SEARCH:PATTERN:CLOCK:
HYSTERESIS 4.0
:SEARCH:PATTERN:CLOCK:HYSTERESIS?
-> :SEARCH:PATTERN:CLOCK:
HYSTERESIS 4.00
```

Description When :SEARch:PATTern:CLOCk:SOURce is set to NONE, an error occurs.

:SEARch:PATTern:CLOCk:LEVel

Function Sets or queries the search level of the clock channel for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:CLOCk:LEVel {<NRf>|
<Voltage>}
:SEARch:PATTern:CLOCk:LEVel?
```
<NRf> = See the Features Guide for this information.

Example
```
:SEARCH:PATTERN:CLOCK:LEVEL
:SEARCH:PATTERN:CLOCK:LEVEL?
-> :SEARCH:PATTERN:CLOCK:LEVEL
```

Description
- When :SEARch:PATTern:CLOCk:SOURce is set to NONE, an error occurs.
- When a current probe is in use, this command does not set or query the <Current> value.

:SEARch:PATTern:CLOCk:SLOPe

Function Sets or queries the slope of the clock channel for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:CLOCk:SLOPe {BOTH|
FALL|RISE}
:SEARch:PATTern:CLOCk:SLOPe?
```

Example
```
:SEARCH:PATTERN:CLOCK:SLOPE RISE
:SEARCH:PATTERN:CLOCK:SLOPE?
-> :SEARCH:PATTERN:CLOCK:SLOPE RISE
```

Description When :SEARch:PATTern:CLOCk:SOURce is set to NONE, an error occurs.

:SEARch:PATTern:CLOCk:SOURce

Function Sets or queries the source waveform of the clock channel for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:CLOCk:SOURce {<NRf>|BIT<x>|NONE|MATH<x>}
:SEARch:PATTern:CLOCk:SOURce?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:SEARCH:PATTERN:CLOCK:SOURCE 1
:SEARCH:PATTERN:CLOCK:SOURCE?
-> :SEARCH:PATTERN:CLOCK:SOURCE 1
```

:SEARch:PATTern:CONDition

Function Sets or queries the achievement condition of the source waveform for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:CONDition {ENTer|EXIT|FALSe|TRUE}
:SEARch:PATTern:CONDition?
```

Example
```
:SEARCH:PATTERN:CONDITION ENTER
:SEARCH:PATTERN:CONDITION?
-> :SEARCH:PATTERN:CONDITION ENTER
```

:SEARch:PATTern:LOGic

Function Sets or queries the logic of the source waveform for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:LOGic {AND|OR}
:SEARch:PATTern:LOGic?
```

Example
```
:SEARCH:PATTERN:LOGIC AND
:SEARCH:PATTERN:LOGIC?
-> :SEARCH:PATTERN:LOGIC AND
```

:SEARch:PATTern:TIME<x>

Function Sets or queries the reference time of the source waveform for the combination (pattern) search of multiple inputs.

Syntax
```
:SEARch:PATTern:TIME<x> {<Time>}
:SEARch:PATTern:TIME<x>?
```
<x> = 1 or 2
<Time> = 1ns to 10s

Example
```
:SEARCH:PATTERN:TIME1 1S
:SEARCH:PATTERN:TIME1?
-> :SEARCH:PATTERN:TIME1 1.000E+00
```

:SEARch:PATTern:TQUalify

Function Sets or queries the time condition of the source waveform for the combination (pattern) of multiple inputs.

Syntax
```
:SEARch:PATTern:TQUalify {INSide|LTHan|MTHan|OUTSide|TIMeout}
:SEARch:PATTern:TQUalify?
```

Example
```
:SEARCH:PATTERN:TQUALIFY INSIDE
:SEARCH:PATTERN:TQUALIFY?
-> :SEARCH:PATTERN:TQUALIFY INSIDE
```

:SEARch:PULSe?

Function Queries all pulse width search settings.

Syntax
```
:SEARch:PULSe?
```

:SEARch:PULSe:HYSTeresis

Function Sets or queries the pulse width search level hysteresis.

Syntax
```
:SEARch:PULSe:HYSTeresis {<NRf>}
:SEARch:PULSe:HYSTeresis?
```
<NRf> = 0.3 to 8.0 div (in 0.1 steps)

Example
```
:SEARCH:PULSE:HYSTERESIS 1
:SEARCH:PULSE:HYSTERESIS?
-> :SEARCH:PULSE:HYSTERESIS 1.0
```

:SEARch:PULSe:LEVel

Function Sets or queries the pulse width search level.

Syntax
```
:SEARch:PULSe:LEVel {<NRf>|<Voltage>}
:SEARch:PULSe:LEVel?
```
<NRf> = See the Features Guide for this information.

Example
```
:SEARCH:PULSE:LEVEL 1V
:SEARCH:PULSE:LEVEL?
-> :SEARCH:PULSE:LEVEL 1.000E+00
```

Description When a current probe is in use, this command does not set or query the <Current> value.

:SEARch:PULSe:POLarity

Function Sets or queries the pulse width search level polarity.

Syntax
```
:SEARch:PULSe:POLarity {HIGH|LOW}
:SEARch:PULSe:POLarity?
```

Example
```
:SEARCH:PULSE:POLARITY HIGH
:SEARCH:PULSE:POLARITY?
-> :SEARCH:PULSE:POLARITY HIGH
```

:SEARch:PULSe:SOURce

Function Sets or queries the source waveform of pulse width search.

Syntax
```
:SEARch:PULSe:SOURce {<NRf>|BIT<x>|MATH<x>|NONE}
:SEARch:PULSe:SOURce?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:SEARCH:PULSE:SOURCE 1
:SEARCH:PULSE:SOURCE?
-> :SEARCH:PULSE:SOURCE 1
```

:SEARch:PULSe:TIME<x>

Function Sets or queries the reference time of pulse width search.

Syntax
```
:SEARch:PULSe:TIME<x> {<Time>}
:SEARch:PULSe:TIME<x>?
```
<x> = 1 or 2
<Time> = 1ns to 10s

Example
```
:SEARCH:PULSE:TIME1 1S
:SEARCH:PULSE:TIME1?
-> :SEARCH:PULSE:TIME1 1.000E+00
```

:SEARch:PULSe:TQUalify

Function Sets or queries the time condition for pulse width searching.

Syntax
```
:SEARch:PULSe:TQUalify {INSide|LTHan
|MTHan|OUTSide|TIMeout}
:SEARch:PULSe:TQUalify?
```

Example
```
:SEARCH:PULSE:TQUALIFY INSIDE
:SEARCH:PULSE:TQUALIFY?
-> :SEARCH:PULSE:TQUALIFY INSIDE
```

:SEARch:SELect

Function Queries the setting of the detected point to show in the zoom window and the zoom position of that point.

Syntax
```
:SEARch:SELect {<NRf>|MAXimum}
:SEARch:SELect?
```
<NRf> = 0 to 49999

Example
```
:SEARCH:SELECT 1
:SEARCH:SELECT?
-> :SEARCH:SELECT 1.50000000
```

Description If there are no detected points, the instrument returns “NAN.”

:SEARch:SELect? MAXimum

Function Queries the total number of detected points.

Syntax
```
:SEARch:SELect? {MAXimum}
```

Example
```
:SEARCH:SELECT? MAXIMUM
-> :SEARCH:SELECT 100
```

Description If there are no detected points, the instrument returns “NAN.”

:SEARch:SKIP?

Function Queries all skip mode settings.

Syntax
```
:SEARch:SKIP?
```

:SEARch:SKIP:DECimation

Function Sets or queries the decimation detection of skip mode.

Syntax
```
:SEARch:SKIP:DECimation {<NRf>}
:SEARch:SKIP:DECimation?
```
<NRf> = 1 to 9999

Example
```
:SEARCH:SKIP:DECIMATION 1
:SEARCH:SKIP:DECIMATION?
-> :SEARCH:SKIP:DECIMATION 1
```

:SEARch:SKIP:HOLDoff

Function Sets or queries the hold-off detection.

Syntax
```
:SEARch:SKIP:HOLDoff {<Time>}
:SEARch:SKIP:HOLDoff?
```
<Time> = 0.1ns to 1s

Example
```
:SEARCH:SKIP:HOLDOFF 0S
:SEARCH:SKIP:HOLDOFF?
-> :SEARCH:SKIP:HOLDOFF 0.000E+00
```

:SEARch:SKIP:MODE

Function Sets or queries the mode of skip mode.

Syntax
```
:SEARch:SKIP:MODE {DECimation|
HOLDoff|OFF}
:SEARch:SKIP:MODE?
```

Example
```
:SEARCH:SKIP:MODE DECIMATION
:SEARCH:SKIP:MODE?
-> :SEARCH:SKIP:MODE DECIMATION
```

:SEARch:SPOint

Function Sets or queries the search start position.

Syntax
```
:SEARch:SPOint {<NRf>}
:SEARch:SPOint?
```
<NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:SEARCH:SPOINT -1
:SEARCH:SPOINT?
-> :SEARCH:SPOINT -1.000E+00
```

:SEARch:TIMeout?

Function Queries all timeout period search settings.

Syntax
```
:SEARch:TIMeout?
```

:SEARch:TIMeout:POLarity

Function Sets or queries the timeout period search polarity.

Syntax
```
:SEARch:TIMeout:POLarity {NEGative|
POSitive}
:SEARch:TIMeout:POLarity?
```

Example
```
:SEARCH:TIMEOUT:POLARITY NRGATIVE
:SEARCH:TIMEOUT:POLARITY?
-> :SEARCH:TIMEOUT:POLARITY NRGATIVE
```

:SEARch:TIMeout:SOURce

Function Sets or queries the timeout period search source waveform.

Syntax
```
:SEARch:TIMeout:SOURce {<NRf>|
MATH<x>|BIT<x>}
:SEARch:TIMeout:SOURce?
```
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:SEARCH:TIMEOUT:SOURCE 1
:SEARCH:TIMEOUT:SOURCE?
-> :SEARCH:TIMEOUT:SOURCE 1
```

:SEARch:TIMeout:TIME

Function Sets or queries the timeout period of the timeout period search.

Syntax
```
:SEARch:TIMeout:TIME <Time>
:SEARch:TIMeout:TIME?
```
<Time> = 1 ns to 10 s

Example
```
:SEARCH:TIMEOUT:TIME 1
:SEARCH:TIMEOUT:TIME?
-> :SEARCH:TIMEOUT:TIME 1
```

:SEARch:TWINdow

Function Sets or queries the zoom window for showing the detected area.

Syntax
```
:SEARch:TWINdow {Z1|Z2}
:SEARch:TWINdow?
```

Example
```
:SEARCH:TWINDOW Z1
:SEARCH:TWINDOW?
-> :SEARCH:TWINDOW Z1
```

:SEARch:TYPE

Function Sets or queries the search type.

Syntax
```
:SEARch:TYPE {EDGE|PATTern|PULSe|TIM
eout}
:SEARch:TYPE?
```

Example
```
:SEARCH:TYPE EDGE
:SEARCH:TYPE?
-> :SEARCH:TYPE EDGE
```

5.24 SERialbus Group

The following table indicates how the communication commands for serial bus types correspond to the serial bus types that appear when you press a panel key.

Serial Bus Type	Communication Command	Setup Menu	
		Panel Key	Serial Bus Type
CAN	CAN*	SHIFT+SEARCH (SERIAL BUS) Key	CAN
CAN FD	CANFD*		CAN FD
CXPI	CXPI*		CXPI
FlexRay	FLEXray*		FlexRay
I2C	I2C*		I2C
LIN	LIN*		LIN
SENT	SENT*		SENT
User-defined	SPATtern		User Define
SPI	SPI*		SPI
UART	UART*		UART

* CAN, CANFD, CXPI, FLEXray, I2C, LIN, SENT, SPI, and UART are serial bus options available only to 4-channel models. These commands cannot be used on 2-channel models or 4-channel models without these options.

:SERialbus<x>?

Function Queries all serial bus signal analysis and search settings.

Syntax `:SERialbus<x>?`
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:ASETup:ABORt

Function Aborts the serial bus signal auto setup.

Syntax `:SERialbus<x>:ASETup:ABORt`
<x> = 1 to 4

Example `:SERIALBUS1:ASETUP:ABORT`

Description This command cannot be used on 2-channel models.

:SERialbus<x>:ASETup:EXECute

Function Execute an serial bus signal auto setup.

Syntax `:SERialbus<x>:ASETup:EXECute`
<x> = 1 to 4

Example `:SERIALBUS1:ASETUP:EXECUTE`

Description This command cannot be used on 2-channel models.

:SERialbus<x>:CAN?

Function Queries all CAN bus signal settings.

Syntax `:SERialbus<x>:CAN?`
<x> = 1 to 4

:SERialbus<x>:CAN:ANALyze?

Function Queries all CAN bus signal analysis settings.

Syntax `:SERialbus<x>:CAN:ANALyze?`
<x> = 1 to 4

:SERialbus<x>:CAN[:ANALyze]:SETup?

Function Queries all CAN bus signal analysis bus settings.

Syntax `:SERialbus<x>:CAN[:ANALyze]:SETup?`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:ANALYZE:SETUP?`
`-> :SERIALBUS1:CAN:ANALYZE:SETUP`

:SERialbus<x>:CAN[:ANALyze]:SETup:BRATe

Function Sets or queries the CAN bus signal analysis bit rate (data transfer rate).

Syntax `:SERialbus<x>:CAN[:ANALyze]:SETup:BRATe {<NRf>|USER,<NRf>}`
`:SERialbus<x>:CAN[:ANALyze]:SETup:BRATe?`
<x> = 1 to 4
<NRf> = 33300, 83300, 125000, 250000, 500000, 1000000
USER <NRf> = See the Features Guide for this information.

Example `:SERIALBUS1:CAN:ANALYZE:SETUP:BRATE 83300`
`:SERIALBUS1:CAN:ANALYZE:SETUP:BRATE?`
`-> :SERIALBUS1:CAN:ANALYZE:SETUP:BRATE 83300`

:SERialbus<x>:CAN[:ANALyze]:SETup:RECessive

Function Sets or queries the CAN bus signal analysis recessive level.

Syntax
```
:SERialbus<x>:CAN[:ANALyze]:SETup:RECessive {HIGH|LOW}
:SERialbus<x>:CAN[:ANALyze]:SETup:RECessive?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:ANALYZE:SETUP:RECESSIVE HIGH
:SERIALBUS1:CAN:ANALYZE:SETUP:RECESSIVE?
-> :SERIALBUS1:CAN:ANALYZE:SETUP:RECESSIVE HIGH
```

:SERialbus<x>:CAN[:ANALyze]:SETup:SOURce

Function Sets or queries the CAN bus signal analysis source.

Syntax
```
:SERialbus<x>:CAN[:ANALyze]:SETup:SOURce {<NRf>|MATH<y>}
:SERialbus<x>:CAN[:ANALyze]:SETup:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:CAN:ANALYZE:SETUP:SOURCE 1
:SERIALBUS1:CAN:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:CAN:ANALYZE:SETUP:SOURCE 1
```

:SERialbus<x>:CAN[:ANALyze]:SETup:SPOint

Function Sets or queries the CAN bus signal analysis sample point.

Syntax
```
:SERialbus<x>:CAN[:ANALyze]:SETup:SPOint {<NRf>}
:SERialbus<x>:CAN[:ANALyze]:SETup:SPOint?
```
<x> = 1 to 4
<NRf> = 18.8 to 90.6

Example
```
:SERIALBUS1:CAN:ANALYZE:SETUP:SPOINT 18.8
:SERIALBUS1:CAN:ANALYZE:SETUP:SPOINT?
-> :SERIALBUS1:CAN:ANALYZE:SETUP:SPOINT 18.8
```

:SERialbus<x>:CAN:DETail?

Function Queries all CAN bus signal analysis result list settings.

Syntax
```
:SERialbus<x>:CAN:DETail?
```
<x> = 1 to 4

:SERialbus<x>:CAN:DETail:DISPlay

Function Sets or queries the display mode for the CAN bus signal analysis result list.

Syntax
```
:SERialbus<x>:CAN:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:CAN:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:DETAIL:DISPLAY FULL
:SERIALBUS1:CAN:DETAIL:DISPLAY?
-> :SERIALBUS1:CAN:DETAIL:DISPLAY FULL
```

:SERialbus<x>:CAN:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the CAN bus signal analysis result list.

Syntax
```
:SERialbus<x>:CAN:DETail:LIST:ITEM?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:DETAIL:LIST:ITEM?
-> :SERIALBUS1:CAN:DETAIL:LIST:ITEM "No.,Time(ms),Frame,ID,DLC,Data,CRC,Ack,Information"
```

:SERialbus<x>:CAN:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the CAN bus signal analysis result list.

Syntax
```
:SERialbus<x>:CAN:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:CAN:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:CAN:DETAIL:LIST:VALUE "0,0.0750,Data,012,1,FE,2263,Y,,"
```

:SERialbus<x>:CAN:SEARch?

Function Queries all CAN bus signal search settings.

Syntax
```
:SERialbus<x>:CAN:SEARch?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:ABORt

Function Aborts the CAN bus signal search.

Syntax
```
:SERialbus<x>:CAN:SEARch:ABORt
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:ABORT
```

:SERialbus<x>:CAN:SEARch:EXECute

Function Executes a CAN bus signal search.

Syntax `:SERialbus<x>:CAN:SEARch:EXECute`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:SEARCH:EXECUTE`

:SERialbus<x>:CAN:SEARch:FJUMp:ACK

Function Jumps to the ACK field in the CAN bus signal search result.

Syntax `:SERialbus<x>:CAN:SEARch:FJUMp:ACK`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:SEARCH:FJUMP:ACK`

:SERialbus<x>:CAN:SEARch:FJUMp:CONTrol

Function Jumps to the control field in the CAN bus signal search result.

Syntax `:SERialbus<x>:CAN:SEARch:FJUMp:CONTrol`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:SEARCH:FJUMP:CONTROL`

:SERialbus<x>:CAN:SEARch:FJUMp:CRC

Function Jumps to the CRC field in the CAN bus signal search result.

Syntax `:SERialbus<x>:CAN:SEARch:FJUMp:CRC`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:SEARCH:FJUMP:CRC`

:SERialbus<x>:CAN:SEARch:FJUMp:DATA

Function Jumps to the data field in the CAN bus signal search result.

Syntax `:SERialbus<x>:CAN:SEARch:FJUMp:DATA`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:SEARCH:FJUMP:DATA`

:SERialbus<x>:CAN:SEARch:FJUMp:IDENtifier

Function Jumps to the identifier in the CAN bus signal search result.

Syntax `:SERialbus<x>:CAN:SEARch:FJUMp:IDENtifier`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:SEARCH:FJUMP:IDENTIFIER`

:SERialbus<x>:CAN:SEARch:FJUMp:SOF

Function Jumps to the SOF in the CAN bus signal search result.

Syntax `:SERialbus<x>:CAN:SEARch:FJUMp:SOF`
<x> = 1 to 4

Example `:SERIALBUS1:CAN:SEARCH:FJUMP:SOF`

:SERialbus<x>:CAN:SEARch:SELect

Function Sets which detected point to display in the CAN bus signal search zoom window and queries the zoom position of the detected point.

Syntax `:SERialbus<x>:CAN:SEARch:SELect {<NRf>|MAXimum}`
`:SERialbus<x>:CAN:SEARch:SELect?`
<x> = 1 to 4
<NRf> = 0 to 49999

Example
```
:SERIALBUS1:CAN:SEARCH:SELECT 1
:SERIALBUS1:CAN:SEARCH:SELECT?
-> :SERIALBUS1:CAN:SEARCH:SELECT 1.50000000
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:CAN:SEARch:SELect? MAXimum

Function Queries the number of detected points in the CAN bus signal search.

Syntax `:SERialbus<x>:CAN:SEARch:SELect? {MAXimum}`
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SELECT? MAXIMUM
-> :SERIALBUS1:CAN:SEARCH:SELECT 100
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:CAN:SEARch:SETup?

Function Queries all CAN bus signal search condition settings.

Syntax `:SERialbus<x>:CAN:SEARch:SETup?`
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup:EFRame?

Function Queries all CAN bus signal search error settings.

Syntax `:SERialbus<x>:CAN:SEARch:SETup:EFRame?`
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup:EFRame[:MODE]

Function Sets or queries the CAN bus signal search error frame setting.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup:EFRame[:MODE] {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup:EFRame[:MODE]?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:MODE ON
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:MODE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:MODE 1
```

:SERialbus<x>:CAN:SEARch:SETup:EFRame:CRC

Function Sets or queries the CAN bus signal search CRC error setting.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup:EFRame:CRC {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup:EFRame:CRC?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:CRC ON
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:CRC?
-> :SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:CRC 1
```

:SERialbus<x>:CAN:SEARch:SETup:EFRame:STUFF

Function Sets or queries the CAN bus signal search stuff error setting.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup:EFRame:STUFF {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup:EFRame:STUFF?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:STUFF ON
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:STUFF?
-> :SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:STUFF 1
```

:SERialbus<x>:CAN:SEARch:SETup:IDData?

Function Queries all ID and data condition settings for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup:IDData?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK?

Function Queries all ID and data condition ACK settings for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:MODE

Function Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:MODE {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:ACK:MODE ON
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:ACK:MODE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:ACK:MODE 1
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:TYPE

Function Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:TYPE {ACK|ACKBoth|NONack}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:TYPE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:ACK:TYPE ACK
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:ACK:TYPE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:ACK:TYPE ACK
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA?

Function Queries all data settings for the ID and data conditions for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:CONDition

Function Sets or queries the data comparison condition, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DECimal<y>

Function Sets a reference value, which is one of the ID and data conditions, for CAN bus signal searching in decimal notation.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DECimal<y> {<NRf>,<NRf>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DECIMAL1?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DLC

Function Sets or queries the number of significant bytes (DLC), which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DLC {<NRf>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DLC?
```
<x> = 1 to 4
<NRf> = 0 to 8

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DLC 0
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DLC?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DLC 0
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa

Function Sets a reference value, which is one of the ID and data conditions, for CAN bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DLC setting.

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern

Function Sets or queries the data, which is one of the ID and data conditions, for CAN bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern {<String>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN ?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DLC setting.

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:ENDian

Function Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:ENDian {BIG|LITTle}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:ENDian?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:ENDIAN?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:MODE

Function Sets or queries the data condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:MODE {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:MODE ON
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:MODE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:MODE 1
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:MSBLsb

Function Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:MSBLsb?
```
<x> = 1 to 4
<NRf> = 0 to 63

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:MSBLSB 1,0
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:MSBLSB?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:MSBLSB 1,0
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:SIGN

Function Sets or queries the sign, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:SIGN {SIGN|UNSign}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:SIGN?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:SIGN?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier?

Function Queries the ID, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:HEXa

Function Sets an ID, which is one of the ID and data conditions, for CAN bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:HEXA "1AB"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MFORmat setting.

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MFORmat {EXTended|STANdard}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:MFORMAT EXTENDED
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:MFORMAT?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:MFORMAT EXTENDED
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MODE

Function Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MODE {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:MODE ON
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:MODE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:MODE 1
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PATTern

Function Sets or queries the ID condition, which is one of the ID and data conditions, for CAN bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PATTern{<String>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PATTERN "11100001111"
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PATTERN?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PATTERN "11100001111"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MFORmat setting.

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PFORmat {MESSage|PATTern}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT PATTERN
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT PATTERN
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal?

Function Queries the message signal, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:MESSage:ITEM

Function Sets the message item, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:MESSage:ITEM{<String>}
```
<x> = 1 to 4

<String> = Up to 32 characters

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:MESSAGE:ITEM "TEST"
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SELect

Function Sets or queries the message signal condition, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SELect {MESSage|SIGNal}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SELect?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SELECT MESSAGE
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SELECT?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SELECT MESSAGE
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal?

Function Queries the signal, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition

Function Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION EQUAL
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION EQUAL
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:DECimal<y>

Function Sets a reference signal value, which is one of the ID and data conditions, for CAN bus signal searching in decimal notation.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:DECimal<y> {<NRf>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:DECIMAL1 1
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:DECIMAL1?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:DECIMAL1 1.000E+00
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:ITEM

Function Sets the signal item, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:ITEM {<String>,<String>}
```
<x> = 1 to 4
<String> = Up to 32 characters

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:ITEM "Sig_Test","Mess_Test"
```

Description Set the signal item first and then the message item.

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:RTR

Function Sets or queries the RTR value, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:RTR {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:RTR?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:RTR ON
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:RTR?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:RTR 1
```

:SERialbus<x>:CAN:SEARch:SETup:MODE

Function Sets or queries the CAN bus signal search mode setting.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup:MODE {EFRame|IDData|SOF}
:SERialbus<x>:CAN:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:MODE EFRAME
:SERIALBUS1:CAN:SEARCH:SETUP:MODE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:MODE EFRAME
```

:SERialbus<x>:CANFD?

Function Queries all CAN FD bus signal settings.

Syntax
```
:SERialbus<x>:CANFD?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:ANALyze?

Function Queries all CAN FD bus signal analysis settings.

Syntax
```
:SERialbus<x>:CANFD:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:CANFD[:ANALyze]:SETup?

Function Queries all CAN FD bus signal analysis bus settings.

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP
```

:SERialbus<x>:CANFD[:ANALyze]:SETup:BRATe

Function Sets or queries the CAN FD bus signal analysis bit rate (data transfer rate).

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup:BRATe {<NRf>|USER,<NRf>}
:SERialbus<x>:CANFD[:ANALyze]:SETup:BRATe?
```
<x> = 1 to 4
<NRf> = 250000, 500000, 1000000
USER <NRf> = 20000 to 1000000 (resolution: 100)

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP:BRATE 1000000
:SERIALBUS1:CANFD:ANALYZE:SETUP:BRATE?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP:BRATE 1000000
```

:SERialbus<x>:CANFD[:ANALyze]:SETup:DBRate

Function Sets or queries the CAN FD bus signal analysis data bit rate (data transfer rate of data phase).

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup:DBRate {<NRf>|USER,<NRf>}
:SERialbus<x>:CANFD[:ANALyze]:SETup:DBRate?
```
<x> = 1 to 4
<NRf> = 500000, 1000000, 2000000, 4000000, 5000000, 8000000
USER <NRf> = 250000 to 10000000 (resolution: 100)

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP:DBRATE 5000000
:SERIALBUS1:CANFD:ANALYZE:SETUP:DBRATE?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP:DBRATE 5000000
```

:SERialbus<x>:CANFD[:ANALyze]:SETup:DSPoint

Function Sets or queries the CAN FD bus signal analysis data phase sample point.

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup:DSPoint {<NRf>}
:SERialbus<x>:CANFD[:ANALyze]:SETup:DSPoint?
```
<x> = 1 to 4
<NRf> = 18.8 to 90.6 (resolution: 0.1)

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP:DSPOINT 75
:SERIALBUS1:CANFD:ANALYZE:SETUP:DSPOINT?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP:DSPOINT 75.00000000
```

:SERialbus<x>:CANFD[:ANALyze]:SETup:FDSTandard

Function Sets or queries whether the CAN FD bus signal to be analyzed is an ISO standard signal.

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup:FDSTandard {ISO|NISO}
:SERialbus<x>:CANFD[:ANALyze]:SETup:FDSTandard?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP:FDSTANDARD ISO
:SERIALBUS1:CANFD:ANALYZE:SETUP:FDSTANDARD?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP:FDSTANDARD ISO
```

:SERialbus<x>:CANFD[:ANALyze]:SETup:RECessive

Function Sets or queries the CAN FD bus signal analysis recessive level.

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup:RECessive {HIGH|LOW}
:SERialbus<x>:CANFD[:ANALyze]:SETup:RECessive?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP:RECESSIVE HIGH
:SERIALBUS1:CANFD:ANALYZE:SETUP:RECESSIVE?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP:RECESSIVE HIGH
```

:SERialbus<x>:CANFD[:ANALyze]:SETup:SOURce

Function Sets or queries the CAN FD bus signal analysis source.

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup:SOURce {<NRf>|MATH<y>}
:SERialbus<x>:CANFD[:ANALyze]:SETup:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP:SOURCE 1
:SERIALBUS1:CANFD:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP:SOURCE 1
```

:SERialbus<x>:CANFD[:ANALyze]:SETup:SPOint

Function Sets or queries the CAN FD bus signal analysis sample point.

Syntax
```
:SERialbus<x>:CANFD[:ANALyze]:SETup:SPOint {<NRf>}
:SERialbus<x>:CANFD[:ANALyze]:SETup:SPOint?
```
<x> = 1 to 4
<NRf> = 18.8 to 90.6 (resolution: 0.1)

Example
```
:SERIALBUS1:CANFD:ANALYZE:SETUP:SPOINT 75
:SERIALBUS1:CANFD:ANALYZE:SETUP:SPOINT?
-> :SERIALBUS1:CANFD:ANALYZE:SETUP:SPOINT 75
```

:SERialbus<x>:CANFD:DETail?

Function Queries all CAN FD bus signal analysis result list settings.

Syntax :SERialbus<x>:CANFD:DETail?
<x> = 1 to 4

:SERialbus<x>:CANFD:DETail:DISPlay

Function Sets or queries the display mode for the CAN FD bus signal analysis result list.

Syntax :SERialbus<x>:CANFD:DETail:DISPl ay {FULL|LOWer|UPPer}
:SERialbus<x>:CANFD:DETail:DISPlay
<x> = 1 to 4

Example :SERIALBUS1:CANFD:DETAIL:DISPL AY FULL
:SERIALBUS1:CANFD:DETAIL:DISPLAY?
-> :SERIALBUS1:CANFD:DETAIL:DISPL AY FULL

:SERialbus<x>:CANFD:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the CAN FD bus signal analysis result list.

Syntax :SERialbus<x>:CANFD:DETail:LIST:IT EM?
<x> = 1 to 4

Example :SERIALBUS1:CANFD:DETAIL:LIST:ITEM?
-> :SERIALBUS1:CANFD:DETAIL:LIST:IT EM "No.,Time(ms),Frame,ID,DLC,Data,C RC,Ack,Information"

:SERialbus<x>:CANFD:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the CAN FD bus signal analysis result list.

Syntax :SERialbus<x>:CANFD:DETail:LIST:VAL ue? {<NRf>|MAXimum|MINimum}
<x> = 1 to 4
<NRf> = –49999 to 49999

Example :SERIALBUS1:CANFD:DETAIL:LIST:VAL UE? 0
-> :SERIALBUS1:CANFD:DETAIL:LIST:VAL UE "0,0.0750,Data,012,1,FE,2263,Y,,"

:SERialbus<x>:CANFD:SEARch?

Function Queries all CAN FD bus signal search settings.

Syntax :SERialbus<x>:CANFD:SEARch?
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:ABORt

Function Aborts the CAN FD bus signal search.

Syntax :SERialbus<x>:CANFD:SEARch:ABORt
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:ABORT

:SERialbus<x>:CANFD:SEARch:EXECute

Function Executes a CAN FD bus signal search.

Syntax :SERialbus<x>:CANFD:SEARch:EXECute
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:EXECUTE

:SERialbus<x>:CANFD:SEARch:FJUMp:ACK

Function Jumps to the ACKField in the CAN FD bus signal search result.
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:ACK

:SERialbus<x>:CANFD:SEARch:FJUMp:CONTrol

Function Jumps to the control field in the CAN FD bus signal search result.

Syntax :SERialbus<x>:CANFD:SEARch:FJUMp:CON Trol
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:CONTR OL

:SERialbus<x>:CANFD:SEARch:FJUMp:CRC

Function Jumps to the CRC field in the CAN FD bus signal search result.

Syntax :SERialbus<x>:CANFD:SEARch:FJUMp:CRC
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:CRC

:SERialbus<x>:CANFD:SEARch:FJUMp:DATA

Function Jumps to the data field in the CAN FD bus signal search result.

Syntax :SERialbus<x>:CANFD:SEARch:FJUMp:DA TA
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:DATA

:SERialbus<x>:CANFD:SEARch:FJUMp:IDENtifier

Function Jumps to the identifier in the CAN FD bus signal search result.

Syntax :SERialbus<x>:CANFD:SEARch:FJUMp:IDE Ntifier
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:IDENT IFIER

:SERialbus<x>:CANFD:SEARch:FJUMp:SOF

Function Jumps to the SOF in the CAN FD bus signal search result.

Syntax :SERialbus<x>:CANFD:SEARch:FJUMp:SOF
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:SOF

:SERialbus<x>:CANFD:SEARch:SELect

Function Sets which detected point to display in the CAN FD bus signal search zoom window and queries the zoom position of the detected point.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SELect {<NRf>|MAXimum}
:SERialbus<x>:CANFD:SEARch:SELect?
```
<x> = 1 to 4
<NRf> = 0 to 49999

Example
```
:SERIALBUS1:CANFD:SEARCH:SELECT 1
:SERIALBUS1:CANFD:SEARCH:SELECT?
-> :SERIALBUS1:CANFD:SEARCH:SELECT 1.50000000
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:CANFD:SEARch:SELect? MAXimum

Function Queries the number of detected points in the CAN FD bus signal search.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SELect? {MAXimum}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SELECT? MAXIMUM
-> :SERIALBUS1:CANFD:SEARCH:SELECT 100
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:CANFD:SEARch:SETup?

Function Queries all CAN FD bus signal search condition settings.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup:EFRame?

Function Queries all CAN FD bus signal search error settings.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRC

Function Sets or queries the CAN FD bus signal search CRC error setting.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRC {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRC?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRC ON
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRC?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRC 1
```

:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor?

Function Queries all CRC error factor settings for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:CRCSequence

Function Sets or queries the CRC sequence (a CRC error factor) for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:CRCSequence {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:CRCSequence?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRCFACTOR:CRCSEQUENCE ON
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRCFACTOR:CRCSEQUENCE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRCFACTOR:CRCSEQUENCE 1
```

:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:SCOunt

Function Sets or queries the Stuff Count (a CRC error factor) for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:SCOunt {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:SCOunt?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRCEFACTOR:SCOUNT ON
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRCEFACTOR:SCOUNT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:CRCEFACTOR:SCOUNT 1
```

:SERialbus<x>:CANFD:SEARch:SETup:EFRame:FSTuff

Function Sets or queries the CAN FD bus signal search fixed stuff error setting.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:FSTuff {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:FSTuff?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:FSTUFF ON
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:FSTUFF?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:FSTUFF 1
```

:SERialbus<x>:CANFD:SEARch:SETup:EFRame[:MODE]

Function Sets or queries the CAN FD bus signal search error frame setting.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame[:MODE] {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup:EFRame[:MODE]?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:MODE ON
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:MODE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:MODE 1
```

:SERialbus<x>:CANFD:SEARch:SETup:EFRame:STUFF

Function Sets or queries the CAN FD bus signal search stuff error setting.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:STUFF {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:STUFF?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:STUFF ON
:SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:STUFF?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:EFRAME:STUFF 1
```

:SERialbus<x>:CANFD:SEARch:SETup:FDF:CONDition

Function Sets or queries the CAN FD bus signal search FDF condition setting.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:FDF:CONDition {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup:FDF:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:FDF:CONDITION ON
:SERIALBUS1:CANFD:SEARCH:SETUP:FDF:CONDITION?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:FDF:CONDITION 1
```

Description When set to ON, CAN FD frames are searched. When set to OFF, CAN frames are searched.

:SERialbus<x>:CANFD:SEARch:SETup:IDData?

Function Queries all ID and data condition settings for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:IDData?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK?

Function Queries all ACK settings for the ID and data conditions for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:MODE

Function Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:MODE {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:ACK:MODE ON
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:ACK:MODE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:ACK:MODE 1
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:TYPE

Function Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:TYPE {ACK|ACKBoth|NONack}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:TYPE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:ACK:TYPE ACK
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:ACK:TYPE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:ACK:TYPE ACK
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA?

Function Queries all data settings for the ID and data conditions for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:BCOunt

Function Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:BCOUNT {<NRf>}
```
<x> = 1 to 4
<NRf> = 0 to 63

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:BCOUNT 4
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:BCOUNT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:BCOUNT 4
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:CONDition

Function Sets or queries the data comparison condition, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DBYTe

Function Sets or queries the number of data bytes, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DBYTe {<NRf>}
```
<x> = 1 to 4
<NRf> = 0 to 8

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DBYTE 2
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DBYTE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DBYTE 2
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DECimal<y>

Function Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in decimal notation.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DECimal<y> {<NRf>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = –9E+18 to 9E+18

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1000
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DECIMAL1?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1.0000000E+03
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa

Function Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DBYTe setting.

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern

Function Sets or queries the data value, which is one of the ID and data conditions, for CAN FD bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern {<String>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DBYTe setting.

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:ENDian

Function Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:ENDian {BIG|LITTle}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:ENDian?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:ENDIAN?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:MODE

Function Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:MODE {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:MODE ON
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:MODE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:MODE 1
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:MSBLsb

Function Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:MSBLsb?
```
<x> = 1 to 4
<NRf> = 0 to 63

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:MSBLSB 15,0
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:MSBLSB?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:MSBLSB 15,0
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:SIGN

Function Sets or queries whether signed or unsigned data format will be used, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:SIGN {SIGN|UNSign}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:SIGN?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:SIGN?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier?

Function Queries all identifier settings for the ID and data conditions for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:HEXa

Function Sets the ID value, which is one of the ID and data conditions, for CAN FD bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:HEXA "1AB"
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:MFORmat {EXTended|STANdard}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:MFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:MFORMAT EXTENDED
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:MFORMAT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:MFORMAT EXTENDED
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:MODE {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:MODE ON
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:MODE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:MODE 1
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:PATTern

Function Sets or queries the ID pattern, which is one of the ID and data conditions, for CAN FD bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:PATTern {<String>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:PATTERN "11100001111"
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:PATTERN?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:PATTERN "11100001111"
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:PFORmat {MESSage|PATTern}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT PATTERN
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT PATTERN
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal?

Function Queries all message and signal settings for the ID and data conditions for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:MESSage:ITEM

Function Sets the message item, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:MESSage:ITEM {<String>}
```
<x> = 1 to 4

<String> = Up to 32 characters

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:MESSAGE:ITEM "TEST"
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SELect

Function Sets or queries the message and signal condition, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:
SETup[:IDData]:MSIGnal:SELe
ct {MESSage|SIGNal}
:SERialbus<x>:CANFD:SEARch:SETup[:ID
Data]:MSIGnal:SELect?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:MSIGNAL:SELECT MESSAGE
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:MSIGNAL:SELECT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:ID
DATA:MSIGNAL:SELECT MESSAGE
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal?

Function Queries all signal settings for the ID and data conditions for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:ID
Data]:MSIGnal:SIGNal?
```
<x> = 1 to 4

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition

Function Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:ID
Data]:MSIGnal:SIGNal:CONDition {EQUa
l|GREater|INSide|LESS|NOTEqual|OUTSi
de}
:SERialbus<x>:CANFD:SEARch:SETup[:ID
Data]:MSIGnal:SIGNal:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:MSIGNAL:SIGNAL:CONDITION EQUAL
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:MSIGNAL:SIGNAL:CONDITION?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:ID
DATA:MSIGNAL:SIGNAL:CONDITION EQUAL
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal:DECimal<y>

Function Sets a signal's reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in decimal notation.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup
[:IDData]:MSIGnal:SIGNal:DECimal
<y> {<NRf>}
:SERialbus<x>:CANFD:SEARch:SETup[:ID
Data]:MSIGnal:SIGNal:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = –9E+18 to 9E+18

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:MSIGNAL:SIGNAL:DECIMAL1 1000
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:MSIGNAL:SIGNAL:DECIMAL1?
-> :SERIALBUS1:CANFD:SEARCH:SET
UP:IDDATA:MSIGNAL:SIGNAL:DECIM
AL1 1.0000000E+03
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal:ITEM

Function Sets the signal item, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SET
up[:IDData]:MSIGnal:SIGNal:IT
EM {<String>,<String>}
```
<x> = 1 to 4
<String> = Up to 32 characters

Example
```
:SERIALBUS1:CANFD:SEARCH:SET
UP:IDDATA:MSIGNAL:SIGNAL:IT
EM "Sig_Test","Mess_Test"
```

Description Set the signal item first and then the message item.

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:RTR

Function Sets or queries the RTR value, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:ID
Data]:RTR {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup[:ID
Data]:RTR?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:RTR ON
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDAT
A:RTR?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:ID
DATA:RTR 1
```

:SERialbus<x>:CANFD:SEARch:SETup:MODE

Function Sets or queries the CAN FD bus signal search mode setting.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup:MODE {EFRame|ESI|FDF|IDData|SOF}
:SERialbus<x>:CANFD:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:MODE EFRAME
:SERIALBUS1:CANFD:SEARCH:SETUP:MODE?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:MODE EFRAME
```

:SERialbus<x>:CXPI?

Function Queries all CXPI bus signal analysis and search settings.

Syntax
```
:SERialbus<x>:CXPI?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:ANALyze?

Function Queries all CXPI bus signal analysis settings.

Syntax
```
:SERialbus<x>:CXPI:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:CXPI[:ANALyze]:SETup?

Function Queries all CXPI bus signal analysis bus settings.

Syntax
```
:SERialbus<x>:CXPI[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:CXPI[:ANALyze]:SETup:BRATe

Function Sets or queries the CXPI bus signal analysis bit rate (data transfer rate).

Syntax
```
:SERialbus<x>:CXPI[:ANALyze]:SETup:BRATe {<NRf>|USER,<NRf>}
:SERialbus<x>:CXPI[:ANALyze]:SETup:BRATe?
```
<x> = 1 to 4
<NRf> = 4800, 9600, 19200
USER <NRf> = 4000 to 50000

Example
```
:SERIALBUS1:CXPI:ANALYZE:SETUP:BRATE 19200
:SERIALBUS1:CXPI:ANALYZE:SETUP:BRATE?
-> :SERIALBUS1:CXPI:ANALYZE:SETUP:BRATE 19200
```

:SERialbus<x>:CXPI[:ANALyze]:SETup:CEDetection

Function Sets or queries the enable/disable condition of counter error detection for CXPI bus signal analysis.

Syntax
```
:SERialbus<x>:CXPI[:ANALyze]:SETup:CEDetection {<Boolean>}
:SERialbus<x>:CXPI[:ANALyze]:SETup:CEDetection?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:ANALYZE:SETUP:CEDETECTION ON
:SERIALBUS1:CXPI:ANALYZE:SETUP:CEDETECTION?
-> :SERIALBUS1:CXPI:ANALYZE:SETUP:CEDETECTION 1
```

:SERialbus<x>:CXPI[:ANALyze]:SETup:CTOLerance

Function Sets or queries the CXPI bus signal analysis clock tolerance.

Syntax
```
:SERialbus<x>:CXPI[:ANALyze]:SETup:CTOLerance {<NRf>}
:SERialbus<x>:CXPI[:ANALyze]:SETup:CTOLerance?
```
<x> = 1 to 4
<NRf> = ±0.5% to ±5.0% (resolution: 0.1%)

Example
```
:SERIALBUS1:CXPI:ANALYZE:SETUP:CTOLERANCE 5
:SERIALBUS1:CXPI:ANALYZE:SETUP:CTOLERANCE?
-> :SERIALBUS1:CXPI:ANALYZE:SETUP:CTOLERANCE 5.00E+00
```

:SERialbus<x>:CXPI[:ANALyze]:SETup:SOURce

Function Sets or queries the CXPI bus signal analysis source.

Syntax
```
:SERialbus<x>:CXPI[:ANALyze]:SETup:SOURce {<NRf>|MATH<y>}
:SERialbus<x>:CXPI[:ANALyze]:SETup:SOURce?
```
<x> = 1 to 4
<y> = 1 to 4
<NRf> = 1 to 4

Example
```
:SERIALBUS1:CXPI:ANALYZE:SETUP:SOURCE 1
:SERIALBUS1:CXPI:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:CXPI:ANALYZE:SETUP:SOURCE 1
```

:SERialbus<x>:CXPI[:ANALyze]:SETup:TSAMple

Function Sets or queries the logic value (1 or 0) determination threshold for CXPI bus signal analysis.

Syntax
```
:SERialbus<x>:CXPI[:ANALyze]:SETup:TSAMple {<NRf>}
```
<x> = 1 to 4
<NRf> = 0.01 Tbit to 0.3 Tbit (resolution: 0.001 Tbit)

Example
```
:SERIALBUS1:CXPI:ANALYZE:SETUP:TSAMPLE 0.04
:SERIALBUS1:CXPI:ANALYZE:SETUP:TSAMPLE?
-> :SERIALBUS1:CXPI:ANALYZE:SETUP:TSAMPLE 0.04
```

Description For details on the logic value (1 or 0) determination threshold, see T Sample of CXPI analysis in the User's Manual.

:SERialbus<x>:CXPI:DETail?

Function Queries all CXPI bus signal analysis result list settings.

Syntax
```
:SERialbus<x>:CXPI:DETail?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:DETail:DISPlay

Function Sets or queries the display mode for the CXPI bus signal analysis result list.

Syntax
```
:SERialbus<x>:CXPI:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:CXPI:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:DETAIL:DISPLAY FULL
:SERIALBUS1:CXPI:DETAIL:DISPLAY?
-> :SERIALBUS1:CXPI:DETAIL:DISPLAY FULL
```

:SERialbus<x>:CXPI:DETail:LIST:ALL?

Function Queries all the data for all analysis numbers in the CXPI bus signal analysis result list.

Syntax
```
:SERialbus<x>:CXPI:DETail:LIST:ALL?
```
<x> = 1 to 4

Example
```
:SERIALBUS<x>:CXPI:DETAIL:LIST:ALL?
-> #8 (8-digit number of bytes)(data sequence)
```

Description All the data for all analysis numbers are returned in <block data> format. The data of each analysis number is separated by ASCII code 0AH.

:SERialbus<x>:CXPI:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the CXPI bus signal analysis result list.

Syntax
```
:SERialbus<x>:CXPI:DETail:LIST:ITEM?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:DETAIL:LIST:ITEM?
-> :SERIALBUS1:CXPI:DETAIL:LIST:ITEM "No.,Time(ms),ID,DLC,W/S,CT,Data,CRC,Information"
```

:SERialbus<x>:CXPI:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the CXPI bus signal analysis result list.

Syntax
```
:SERialbus<x>:CXPI:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4
<NRf> = –9999 to 9999

Example
```
:SERIALBUS1:CXPI:DETAIL:LIST:VALUE? 1
-> "1,4.7228,P3,8,01,0,00 00 00 00 01 03 05 06,FE,,"
```

:SERialbus<x>:CXPI:SEARch?

Function Queries all CXPI bus signal search settings.

Syntax
```
:SERialbus<x>:CXPI:SEARch?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:ABORt

Function Aborts the CXPI bus signal search.

Syntax
```
:SERialbus<x>:CXPI:SEARch:ABORt
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:ABORT
```

:SERialbus<x>:CXPI:SEARch:EXECute

Function Executes a CXPI bus signal search.

Syntax
```
:SERialbus<x>:CXPI:SEARch:EXECute
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:EXECUTE
```

:SERialbus<x>:CXPI:SEARch:SELect

Function Sets which detected point to display in the CXPI bus signal search zoom window and queries the zoom position of the detected point.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SELect {<NRf>|MAXimum}
:SERialbus<x>:CXPI:SEARch:SELect?
```
<x> = 1 to 4
<NRf> = 0 to 9999

Example
```
:SERIALBUS1:CXPI:SEARCH:SELECT 1
:SERIALBUS1:CXPI:SEARCH:SELECT?
-> :SERIALBUS1:CXPI:SEARCH:SELECT 1.50000000
```

Description If there are no detected points, the instrument returns "NAN."

:SERialbus<x>:CXPI:SEARch:SETup?

Function Queries all CXPI bus signal search condition settings.

Syntax :SERialbus<x>:CXPI:SEARch:SETup?

<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup:ERRor?

Function Queries all CXPI bus signal search error settings.

Syntax :SERialbus<x>:CXPI:SEARch:SETup:ERRor?

<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:CLOCk

Function Sets or queries the CXPI bus signal search clock error setting.

Syntax :SERialbus<x>:CXPI:SEARch:SETup:ERRor:CLOCk {<Boolean>}

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:CLOCk?

<x> = 1 to 4

Example :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:CLOCK ON

:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:CLOCK?

-> :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:CLOCK 1

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:COUNter

Function Sets or queries the CXPI bus signal search counter error setting.

Syntax :SERialbus<x>:CXPI:SEARch:SETup:ERRor:COUNter {<Boolean>}

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:COUNter?

<x> = 1 to 4

Example :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:COUNTER ON

:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:COUNTER?

-> :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:COUNTER 1

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:CRC

Function Sets or queries the CXPI bus signal search CRC error setting.

Syntax :SERialbus<x>:CXPI:SEARch:SETup:ERRor:CRC {<Boolean>}

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:CRC?

<x> = 1 to 4

Example :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:CRC ON

:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:CRC?

-> :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:CRC 1

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:DLENgth

Function Sets or queries the CXPI bus signal search data length error setting.

Syntax :SERialbus<x>:CXPI:SEARch:SETup:ERRor:DLENgth {<Boolean>}

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:DLENgth?

<x> = 1 to 4

Example :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:DLENGTH ON

:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:DLENGTH?

-> :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:DLENGTH 1

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:FRAMing

Function Sets or queries the CXPI bus signal search framing error setting.

Syntax :SERialbus<x>:CXPI:SEARch:SETup:ERRor:FRAMing {<Boolean>}

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:FRAMing?

<x> = 1 to 4

Example :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:FRAMING ON

:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:FRAMING?

-> :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:FRAMING 1

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:IBS

Function Sets or queries the CXPI bus signal search IBS error setting.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:IBS {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:IBS?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:IBS ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:IBS?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:IBS 1
```

:SERialbus<x>:CXPI:SEARch:SETup:ERRor:PARity

Function Sets or queries the CXPI bus signal search parity error setting.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:PARity {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:PARity?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:PARITY ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:PARITY?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:PARITY 1
```

:SERialbus<x>:CXPI:SEARch:SETup:IDData?

Function Queries all ID and data condition settings for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:IDData?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA?

Function Queries all data settings for the ID and data conditions for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:BCOunt

Function Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:BCOunt {<NRf>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:BCOunt?
```
<x> = 1 to 4
<NRf> = 0 to 254

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:BCOUNT 1
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:BCOUNT?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:BCOUNT 1
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:CONDition

Function Sets or queries the data comparison condition, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DBYTe

Function Sets or queries the number of data bytes, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DBYTe {<NRf>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DBYTe?
```
<x> = 1 to 4
<NRf> = 0 to 8

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DBYTE 1
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DBYTE?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DBYTE 1
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DECimal<y>

Function Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in decimal notation.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DECimal<y> {<NRf>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = –9E+18 to 9E+18

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DECIMAL1?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1.0000000E+00
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa

Function Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DBYTe setting.

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern

Function Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern {<String>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DBYTe setting.

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:ENDian

Function Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:ENDian {BIG|LITTle}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:ENDian?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:ENDIAN?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:MODE

Function Sets or queries the ID and data condition (enabled/disabled) for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:MODE {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:MODE ON
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:MODE?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:MODE 1
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:MSBLsb

Function Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:MSBLsb?
```
<x> = 1 to 4
<NRf> = 0 to 63

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:MSBLSB 7,0
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:MSBLSB?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:MSBLSB 7,0
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:SIGN

Function Sets or queries the sign, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:SIGN {SIGN|UNSign}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:SIGN?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:SIGN?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation?

Function Queries all frame information settings for the ID and data conditions for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:CT

Function Sets or queries the frame information counter value, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:CT {<NRf>|DONTcare}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:CT?
```
<x> = 1 to 4
<NRf> = 0 to 3

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:CT 3
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:CT?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:CT 3
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:MODE

Function Sets or queries the frame information (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:MODE {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:MODE ON
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:MODE?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:MODE 1
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:SLEEP

Function Sets or queries the frame information sleep bit, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:SLEEP {0|1|X}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:SLEEP?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:SLEEP 1
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:SLEEP?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:SLEEP 1
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:WAKeup

Function Sets or queries the frame information wakeup bit, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:WAKeup {0|1|X}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:WAKeup?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:WAKEUP 1
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:WAKEUP?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:FINFORMATION:WAKEUP 1
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID?

Function Queries all ID settings for the ID and data conditions for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:HEXa

Function Sets an ID, which is one of the ID and data conditions, for CXPI bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:HEXA "1E"
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:MODE

Function Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:MODE {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:MODE ON
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:MODE?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:MODE 1
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PATTern

Function Sets or queries the ID, which is one of the ID and data conditions, for CXPI bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PATTern {<String>}
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:PATTERN "0010000"
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:PATTERN?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:PATTERN "0010000"
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PTYPE

Function Sets or queries the PTYPE comparison condition, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PTYPE {DONTcare|NO|YES}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:PTYPE DONTCARE
:SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:PTYPE?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:IDDATA:ID:PTYPE DONTCARE
```

:SERialbus<x>:CXPI:SEARch:SETup:MODE

Function Sets or queries the CXPI bus signal search type.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:MODE {ERRor|IDData|PTYPE|SOF|WAKeupsleep}
:SERialbus<x>:CXPI:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:MODE SOF
:SERIALBUS1:CXPI:SEARCH:SETUP:MODE?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:MODE SOF
```

:SERialbus<x>:CXPI:SEARch:SETup:PTYPE

Function Queries all CXPI bus signal search PTYPE settings.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:PTYPE?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup:PTYPE:CONDition

Function Sets or queries the CXPI bus signal search PTYPE condition setting.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:PTYPE:CONDition {EPTYPE|NOResponse}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:PTYPE:CONDITION EPTYPE
:SERIALBUS1:CXPI:SEARCH:SETUP:PTYPE:CONDITION?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:PTYPE:CONDITION EPTYPE
```

Description For EPTYPE, all PTYPEs are searched for. For NOResponse, PTYPEs without responses are searched for.

:SERialbus<x>:CXPI:SEARch:SETup:WAKeup sleep?

Function Queries all wakeup and sleep settings for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep?
```
<x> = 1 to 4

:SERialbus<x>:CXPI:SEARch:SETup:WAKeup sleep:SFRame

Function Sets or queries the CXPI bus signal search sleep frame setting.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:SFRame {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:SFRame?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:SFRAME ON
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:SFRAME?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:SFRAME 1
```

:SERialbus<x>:CXPI:SEARch:SETup:WAKeup sleep:SLEEP

Function Sets or queries the CXPI bus signal search sleep (clock unavailable condition) setting.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:SLEEP {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:SLEEP?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:SLEEP ON
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:SLEEP?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:SLEEP 1
```

:SERialbus<x>:CXPI:SEARch:SETup:WAKeup sleep:WAKeup

Function Sets or queries the CXPI bus signal search wakeup (clock available condition) setting.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:WAKeup {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:WAKeup?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:WAKEUP ON
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:WAKEUP?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:WAKEUP 1
```

:SERialbus<x>:CXPI:SEARch:SETup:WAKeup sleep:WPULse

Function Sets or queries the CXPI bus signal search wakeup pulse setting.

Syntax
```
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:WPULse {<Boolean>}
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:WPULse?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:WPULSE ON
:SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:WPULSE?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:WAKEUPSLEEP:WPULSE 1
```

:SERialbus<x>:DECode?

Function Queries all settings related to the decoding of serial bus signals.

Syntax
```
:SERialbus<x>:DECode?
```
<x> = 1 to 4

:SERialbus<x>:DECode[:FORMat]

Function Sets or queries the decode display format of serial bus signals.

Syntax
```
:SERialbus<x>:DECode[:FORMat] {BINary|DECimal|HEXa|ASCii|SYMBol}
:SERialbus<x>:DECode[:FORMat]?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:DECODE:FORMAT BINARY
:SERIALBUS1:DECODE:FORMAT?
-> :SERIALBUS1:DECODE:FORMAT BINARY
```

:SERialbus<x>:DECode:SSCMode

Function Sets or queries the decode display of the start/stop condition of I2C serial bus signals.

Syntax
```
:SERialbus<x>:DECode:SSCMode {<Boolean>}
:SERialbus<x>:DECode:SSCMode?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:DECODE:SSCMODE ON
:SERIALBUS1:DECODE:SSCMODE?
-> :SERIALBUS1:DECODE:SSCMODE 1
```

:SERialbus<x>:DISPlay

Function Sets or queries whether serial bus signal analysis is to be performed.

Syntax
```
:SERialbus<x>:DISPlay {<Boolean>}
:SERialbus<x>:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:DISPLAY ON
:SERIALBUS1:DISPLAY?
-> :SERIALBUS1:DISPLAY 1
```

:SERialbus<x>:FLEXray?

Function Queries all FLEXRAY bus signal analysis and search settings.

Syntax :SERialbus<x>:FLEXray?
<x> = 1 to 4

:SERialbus<x>:FLEXray:ANALyze?

Function Queries all FLEXRAY bus signal analysis settings.

Syntax :SERialbus<x>:FLEXray:ANALyze?
<x> = 1 to 4

:SERialbus<x>:FLEXray[:ANALyze]:SETup?

Function Queries all FLEXRAY bus signal analysis bus settings.

Syntax :SERialbus<x>:FLEXray[:ANALyze]:SETup?
<x> = 1 to 4

:SERialbus<x>:FLEXray[:ANALyze]:SETup:BCHannel

Function Sets or queries the channel bus type for FLEXRAY bus signal analysis.

Syntax :SERialbus<x>:FLEXray[:ANALyze]:SETup:BCHannel {A|B}
:SERialbus<x>:FLEXray[:ANALyze]:SETup:BCHannel?
<x> = 1 to 4

Example :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BCHANNEL A
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BCHANNEL?
-> :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BCHANNEL A

:SERialbus<x>:FLEXray[:ANALyze]:SETup:BRATe

Function Sets or queries the FLEXRAY bus signal analysis bit rate (data transfer rate).

Syntax :SERialbus<x>:FLEXray[:ANALyze]:SETup:BRATe {<NRf>}
:SERialbus<x>:FLEXray[:ANALyze]:SETup:BRATe?
<x> = 1 to 4
<NRf> = 2500000, 5000000, 10000000

Example :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BRATE 5000000
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BRATE?
-> :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BRATE 5000000

:SERialbus<x>:FLEXray[:ANALyze]:SETup:SOURce

Function Sets or queries the FLEXRAY bus signal analysis source.

Syntax :SERialbus<x>:FLEXray[:ANALyze]:SETup:SOURce {<NRf>|MATH<y>}
:SERialbus<x>:FLEXray[:ANALyze]:SETup:SOURce?
<x> = 1 to 4
<NRf> = 1 to 4
<y> = 1 to 4

Example :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:SOURCE 1
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:SOURCE 1

:SERialbus<x>:FLEXray:DETail?

Function Queries all FLEXRAY bus signal analysis result list settings.

Syntax :SERialbus<x>:FLEXray:DETail?
<x> = 1 to 4

:SERialbus<x>:FLEXray:DETail:DISPlay

Function Sets or queries the display mode for the FLEXRAY bus signal analysis result list.

Syntax :SERialbus<x>:FLEXray:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:FLEXray:DETail:DISPlay?
<x> = 1 to 4

Example :SERIALBUS1:FLEXRAY:DETAIL:DISPLAY FULL
:SERIALBUS1:FLEXRAY:DETAIL:DISPLAY?
-> :SERIALBUS1:FLEXRAY:DETAIL:DISPLAY FULL

:SERialbus<x>:FLEXray:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the FLEXRAY bus signal analysis result list.

Syntax :SERialbus<x>:FLEXray:DETail:LIST:ITEM?
<x> = 1 to 4

Example :SERIALBUS1:FLEXRAY:DETAIL:LIST:ITEM?
-> :SERIALBUS1:FLEXRAY:DETAIL:LIST:ITEM "No.,Time(ms),S/D,IND,ID,Len,CC,Data,Information,"

:SERialbus<x>:FLEXray:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the FLEXRAY bus signal analysis result list.

Syntax `:SERialbus<x>:FLEXray:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}`
<x> = 1 to 4
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:FLEXRAY:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:FLEXRAY:DETAIL:LIST:VALUE "0,-0.0001968,S,1111, 4, 4,2,
01 02 03 04 05 06 07 08,,"
```

:SERialbus<x>:FLEXray:SEARch?

Function Queries all FLEXRAY bus signal search settings.

Syntax `:SERialbus<x>:FLEXray:SEARch?`
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:ABORt

Function Aborts the FLEXRAY bus signal Search.

Syntax `:SERialbus<x>:FLEXray:SEARch:ABORt`
<x> = 1 to 4

Example `:SERIALBUS1:FLEXRAY:SEARCH:ABORT`

:SERialbus<x>:FLEXray:SEARch:EXECute

Function Executes a FLEXRAY bus signal Search.

Syntax `:SERialbus<x>:FLEXray:SEARch:EXECute`
<x> = 1 to 4

Example `:SERIALBUS1:FLEXRAY:SEARCH:EXECUTE`

:SERialbus<x>:FLEXray:SEARch:FJUMp:CCOunt

Function Jumps to the cycle count field in the FLEXRAY bus signal search result.

Syntax `:SERialbus<x>:FLEXray:SEARch:FJUMp:CCOunt`
<x> = 1 to 4

Example `:SERIALBUS1:FLEXRAY:SEARCH:FJUMP:CCOUNT`

:SERialbus<x>:FLEXray:SEARch:FJUMp:CRC

Function Jumps to the CRC field in the FLEXRAY bus signal search result.

Syntax `:SERialbus<x>:FLEXray:SEARch:FJUMp:CRC`
<x> = 1 to 4

Example `:SERIALBUS1:FLEXRAY:SEARCH:FJUMP:CRC`

:SERialbus<x>:FLEXray:SEARch:FJUMp:FRAMeid

Function Jumps to the frame ID field in the FLEXRAY bus signal search result.

Syntax `:SERialbus<x>:FLEXray:SEARch:FJUMp:FRAMeid`
<x> = 1 to 4

Example `:SERIALBUS1:FLEXRAY:SEARCH:FJUMP:FRAMEID`

:SERialbus<x>:FLEXray:SEARch:FJUMp:HCRC

Function Jumps to the header CRC field in the FLEXRAY bus signal search result.

Syntax `:SERialbus<x>:FLEXray:SEARch:FJUMp:HCRC`
<x> = 1 to 4

Example `:SERIALBUS1:FLEXRAY:SEARCH:FJUMP:HCRC`

:SERialbus<x>:FLEXray:SEARch:FJUMp:PLENgth

Function Jumps to the payload length field in the FLEXRAY bus signal search result.

Syntax `:SERialbus<x>:FLEXray:SEARch:FJUMp:PLENgth`
<x> = 1 to 4

Example `:SERIALBUS1:FLEXRAY:SEARCH:FJUMP:PLENGTH`

:SERialbus<x>:FLEXray:SEARch:SELect

Function Sets the detected waveform number for a FLEXRAY bus signal search and queries the zoom position corresponding to that number.

Syntax `:SERialbus<x>:FLEXray:SEARch:SELect {<NRf>|MAXimum}`
`:SERialbus<x>:FLEXray:SEARch:SELect?`
<x> = 1 to 4
<NRf> = 0 to 49999

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SELECT 1
:SERIALBUS1:FLEXRAY:SEARCH:SELECT?
-> :SERIALBUS1:FLEXRAY:SEARCH:SELECT 1.50000000
```

:SERialbus<x>:FLEXray:SEARch:SETup?

Function Queries all FLEXRAY bus signal search condition settings.

Syntax `:SERialbus<x>:FLEXray:SEARch:SETup?`
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup:ERRor?

Function Queries all FLEXRAY bus signal search error settings.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:BSS

Function Sets or queries the FLEXRAY bus signal search BSS Error setting.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:BSS {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:BSS?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:BSS ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:BSS?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:BSS 1
```

:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:CRC

Function Sets or queries the FLEXRAY bus signal search CRC Error setting.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:CRC {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:CRC?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:CRC ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:CRC?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:CRC 1
```

:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:FES

Function Sets or queries the FLEXRAY bus signal search FES Error setting.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:FES {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:FES?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:FES ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:FES?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:FES 1
```

:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:HCRC

Function Sets or queries the FLEXRAY bus signal search Header CRC Error setting.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:HCRC {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:HCRC?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:HCRC ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:HCRC?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:ERROR:HCRC 1
```

:SERialbus<x>:FLEXray:SEARch:SETup:IDData?

Function Queries all ID and data condition settings for FLEXRAY bus signal search.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:IDData?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt?

Function Queries all FLEXRAY bus signal search cycle count settings.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition

Function Sets or queries the FLEXRAY bus signal search cycle count data condition.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:CONDITION EQUAL
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:CONDITION?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:CONDITION EQUAL
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt<y>

Function Sets or queries the FLEXRAY bus signal search cycle count setting.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt<y> {<NRf>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 0 to 63

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:COUNT1 10
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:COUNT1?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:COUNT1 10
```

Description
- When :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDitionEQUal|GREater|NOTEqual is specified, use :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt1 to set the value.
- When :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition LESS is specified, use :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt2 to set the value.
- When :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDitionBETWeen|NOTBetween is specified, use :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt1 to set the smaller value and :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt2 to set the larger value.

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:MODE

Function Sets or queries the cycle count mode, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:MODE {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:MODE ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:MODE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:MODE 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA?

Function Queries the data, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:BCOunt

Function Sets or queries the pattern comparison position, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:BCOunt {<NRf>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:BCOunt?
```
<x> = 1 to 4
<NRf> = 0 to 253

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:BCOUNT 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:BCOUNT?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:BCOUNT 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:CONDition

Function Sets or queries the data comparison condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DBYTe

Function Sets or queries the number of data bytes, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DBYTe {<NRf>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DBYTe?
```
<x> = 1 to 4
<NRf> = 1 to 8

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DBYTE 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DBYTE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DBYTE 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DECimal<y>

Function Sets or queries a reference value, which is one of the ID and data conditions, for FLEXRAY bus signal searching in decimal notation.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DECimal<y> {<NRf>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DECIMAL1?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DECIMAL1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa

Function Sets a reference value, which is one of the ID and data conditions, for FLEXRAY bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DPATtern:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DBYTe setting.

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern

Function Sets or queries the data, which is one of the ID and data conditions, for FLEXRAY bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern {<String>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DPATtern:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DBYTe setting.

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:ENDian

Function Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:ENDian {BIG|LITTle}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:ENDian?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:ENDIAN?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MODE

Function Sets or queries the data condition (enabled/disabled), which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MODE {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:MODE ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:MODE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:MODE 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MSBLsb

Function Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MSBLsb?
```
<x> = 1 to 4
<NRf> = 0 to 63

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:MSBLSB 7,0
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:MSBLSB?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:MSBLSB 7,0
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:SIGN

Function Sets or queries the sign, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:SIGN {SIGN|UNSign}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:SIGN?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:SIGN?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID?

Function Queries the frame ID, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:CONDition

Function Sets or queries the frame ID data condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:CONDITION EQUAL
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:CONDITION?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:CONDITION EQUAL
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:ID<y>

Function Sets or queries the frame ID value, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:ID<y> {<NRf>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:ID<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 or 2047

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:ID1 100
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:ID1?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:ID1 100
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:MODE

Function Sets or queries the frame ID condition (enabled/disabled), which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:MODE {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:MODE ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:MODE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:FID:MODE 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator?

Function Queries the indicator, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:MODE

Function Sets or queries the indicator condition (enabled/disabled), which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:MODE {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:MODE ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:MODE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:MODE 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:NFRame

Function Sets or queries the indicator null frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:NFRame {0|1|X}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:NFRame?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:NFRAME 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:NFRAME?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:NFRAME 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:PPReamble

Function Sets or queries the indicator payload preamble, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:PPReamble {0|1|X}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:PPReamble?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:PPREAMBLE 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:PPREAMBLE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:PPREAMBLE 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:STFRame

Function Sets or queries the indicator start frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:STFRame {0|1|X}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:STFRame?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:STFRAME 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:STFRAME?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:STFRAME 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:SYFRame

Function Sets or queries the indicator sync frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:SYFRame {0|1|X}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:SYFRame?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:SYFRAME 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:SYFRAME?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:INDICATOR:SYFRAME 1
```

:SERialbus<x>:FLEXray:SEARch:SETup:MODE

Function Sets or queries the FLEXRAY bus signal search mode setting.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:MODE {ERRor|FSTart|IDData}
:SERialbus<x>:FLEXray:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:MODE FSTART
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:MODE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:MODE FSTART
```

:SERialbus<x>:I2C?

Function Queries all I2C bus signal analysis and search settings.

Syntax
```
:SERialbus<x>:I2C?
```
<x> = 1 to 4

:SERialbus<x>:I2C:ANALyze?

Function Queries all I2C bus signal analysis settings.

Syntax
```
:SERialbus<x>:I2C:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:I2C[:ANALyze]:SETup?

Function Queries all I2C bus signal analysis bus settings.

Syntax
```
:SERialbus<x>:I2C[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk?

Function Queries all I2C bus signal analysis clock settings.

Syntax
```
:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk?
```
<x> = 1 to 4

:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk:SOURce

Function Sets or queries the I2C bus signal analysis clock.

Syntax
```
:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk:SOURce {<NRf>|BIT<y>|MATH<y>}
:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:I2C:ANALYZE:SETUP:CLOCK:SOURCE 1
:SERIALBUS1:I2C:ANALYZE:SETUP:CLOCK:SOURCE?
-> :SERIALBUS1:I2C:ANALYZE:SETUP:CLOCK:SOURCE 1
```

:SERialbus<x>:I2C[:ANALyze]:SETup:DATA?

Function Queries all I2C bus signal analysis data settings.

Syntax
```
:SERialbus<x>:I2C[:ANALyze]:SETup:DATA?
```
<x> = 1 to 4

:SERialbus<x>:I2C[:ANALyze]:SETup:DATA:SOURce

Function Sets or queries the I2C bus signal analysis data.

Syntax
```
:SERialbus<x>:I2C[:ANALyze]:SETup:DATA:SOURce {<NRf>|BIT<y>|MATH<y>}
:SERialbus<x>:I2C[:ANALyze]:SETup:DATA:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:I2C:ANALYZE:SETUP:DATA:SOURCE 1
:SERIALBUS1:I2C:ANALYZE:SETUP:DATA:SOURCE?
-> :SERIALBUS1:I2C:ANALYZE:SETUP:DATA:SOURCE 1
```

:SERialbus<x>:I2C[:ANALyze]:SETup:INCLuderw

Function Sets or queries the enable/disable condition of address R/W for I2C bus signal analysis.

Syntax
```
:SERialbus<x>:I2C[:ANALyze]:SETup:INCLuderw {<Boolean>}
:SERialbus<x>:I2C[:ANALyze]:SETup:INCLuderw?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:ANALYZE:SETUP:INCLUDERW ON
:SERIALBUS1:I2C:ANALYZE:SETUP:INCLUDERW?
-> :SERIALBUS1:I2C:ANALYZE:SETUP:INCLUDERW 1
```

:SERialbus<x>:I2C:DETail?

Function Queries all I2C bus signal analysis result list settings.

Syntax
```
:SERialbus<x>:I2C:DETail?
```
<x> = 1 to 4

:SERialbus<x>:I2C:DETail:DISPlay

Function Sets or queries the display mode for the I2C bus signal analysis result list.

Syntax
```
:SERialbus<x>:I2C:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:I2C:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:DETAIL:DISPLAY FULL
:SERIALBUS1:I2C:DETAIL:DISPLAY?
-> :SERIALBUS1:I2C:DETAIL:DISPLAY FULL
```

:SERialbus<x>:I2C:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the I2C bus signal analysis result list.

Syntax
```
:SERialbus<x>:I2C:DETail:LIST:ITEM?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:DETAIL:LIST:ITEM?
-> :SERIALBUS1:I2C:DETAIL:LIST
:ITEM "No.,Time(ms),1st,2nd,R/W,Data,Information,"
```

:SERialbus<x>:I2C:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the I2C bus signal analysis result list.

Syntax
```
:SERialbus<x>:I2C:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4

<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:I2C:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:I2C:DETAIL:LIST:
VALUE "0,-0.07000,AA*,,W,AE*88*,7-bit,"
```

:SERialbus<x>:I2C:SEARch?

Function Queries all I2C bus signal search settings.

Syntax
```
:SERialbus<x>:I2C:SEARch?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch:ABORt

Function Aborts the I2C bus signal search.

Syntax
```
:SERialbus<x>:I2C:SEARch:ABORt
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:ABORT
```

:SERialbus<x>:I2C:SEARch:EXECute

Function Executes an I2C bus signal search.

Syntax
```
:SERialbus<x>:I2C:SEARch:EXECute
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:EXECUTE
```

:SERialbus<x>:I2C:SEARch:SELect

Function Sets which detected point to display in the I2C bus signal search zoom window and queries the zoom position of the detected point.

Syntax
```
:SERialbus<x>:I2C:SEARch:SELect {<NRf>|MAXimum}
:SERialbus<x>:I2C:SEARch:SELect?
```
<x> = 1 to 4

<NRf> = 0 to 49999

Example
```
:SERIALBUS1:I2C:SEARCH:SELECT 1
:SERIALBUS1:I2C:SEARCH:SELECT?
-> :SERIALBUS1:I2C:SEARCH:SELECT 1.50000000
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:I2C:SEARch:SETup?

Function Queries all I2C bus signal search condition settings.

Syntax
```
:SERialbus<x>:I2C:SEARch:SETup?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess?

Function Queries all I2C bus signal search address pattern settings.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess?

Function Queries all settings related to the address of the I2C bus signal search address pattern.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress?

Function Queries all I2C bus signal search 10-bit address settings.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:DIRection

Function Sets or queries the 10-bit address data direction for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:DIRection {READ|WRITe|X)
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:DIRection?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT10ADDRESS:DIRECTION READ
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT10ADDRESS:DIRECTION?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT10ADDRESS:DIRECTION READ
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:HEXa

Function Sets the 10-bit address for I2C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT10ADDRESS:HEXA "1AB"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:PATTern

Function Sets or queries the 10-bit address for I2C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:PATTern {<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT10ADDRESS:PATTERN "11010111001"
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT10ADDRESS:PATTERN?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT10ADDRESS:PATTERN "11010111001"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]?

Function Queries all I2C bus signal search 7-bit address settings.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:DIRection

Function Sets or queries the 7-bit address data direction for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:DIRection {READ|WRITe|X)
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:DIRection?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7ADDRESS:DIRECTION READ
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7ADDRESS:DIRECTION?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7ADDRESS:DIRECTION READ
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:HEXa

Function Sets the 7-bit address for I2C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7ADDRESS:HEXA "5C"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:PATTern

Function Sets or queries the 7-bit address for I2C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:PATTern {<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7ADDRESS:PATTERN "11100110"
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7ADDRESS:PATTERN?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7ADDRESS:PATTERN "11100110"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub?

Function Queries all I2C bus signal search 7-bit + sub address settings.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit + sub address for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:DIRection

Function Sets or queries the 7-bit + sub address data direction for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:DIRection {READ|WRITe|X}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:DIRection?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:DIRECTION READ
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:DIRECTION?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:DIRECTION READ
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:HEXa

Function Sets the 7-bit address of 7-bit + sub address for I2C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:HEXA "AB"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern

Function Sets or queries the 7-bit address of 7-bit + sub address for I2C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern {<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:PATTERN "00111010"
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:PATTERN?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:PATTERN "00111010"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress?

Function Queries all settings related to the sub address of the 7-bit + sub address for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress:HEXa

Function Sets the sub address of 7-bit + sub address for I2C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:HEXA "EF"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern

Function Sets or queries the sub address of 7-bit + sub address for I2C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern {<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:PATTERN "00111010"
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:PATTERN?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:PATTERN "00111010"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:MODE

Function Sets or queries the address condition enable/disable state for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:MODE {<Boolean>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:MODE ON
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:MODE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:MODE 1
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:TYPE

Function Sets or queries the address condition address format for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:TYPE {BIT10ADdress|BIT7ADdress|BIT7APsub}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:TYPE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:TYPE BIT10ADDRESS
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:TYPE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:TYPE BIT10ADDRESS
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA?

Function Queries all settings related to the data of the I2C bus signal search address pattern.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BCOunt

Function Sets or queries the data pattern comparison position for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BCOunt {<NRf>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BCOunt?
```
<x> = 1 to 4

<NRf> = 0 to 9999

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:BCOUNT 0
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:BCOUNT?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:BCOUNT 0
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BMODe

Function Sets or queries on/off state of the data pattern comparison position for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BMODe {<Boolean>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BMODe?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:BMODE ON
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:BMODE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:BMODE 1
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:CONDition

Function Sets or queries the data judgment method (true/false) for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:CONDition {FALSe|TRUE}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:CONDITION FALSE
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:CONDITION?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:CONDITION FALSE
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DBYTe

Function Sets or queries the number of reference values for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DBYTe {<NRf>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DBYTe?
```
<x> = 1 to 4
<NRf> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:DBYTE 1
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:DBYTE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:DBYTE 1
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DPATtern:HEXa

Function Sets the data condition data for I2C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DPATtern:HEXa {<String>}
```

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:DPATTERN:HEXA "AB"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DBYTe setting.

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DPATtern:PATTern

Function Sets or queries the data condition data for I2C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DPATtern:PATTern {<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DPATtern:PATTern?
```

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:DPATTERN:PATTERN "10001101"
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:DPATTERN:PATTERN?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:DPATTERN:PATTERN "10001101"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DBYTe setting.

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:MODE

Function Sets or queries the data condition enable/disable state for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:MODE {<Boolean>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:MODE ON
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:MODE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:MODE 1
```

:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall?

Function Queries all I2C bus signal search general call settings.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress?

Function Queries all settings related to the 7-bit master address of general call for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress:HEXa

Function Sets the 7-bit master address of general call for I2C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress:HEXa{<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:GENERALCALL:BIT7MADDRESS:HEXA "AB"
```

:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress:PATTern

Function Sets or queries the 7-bit master address of general call for I2C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress:PATTern{<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:GENERALCALL:BIT7MADDRESS:PATTERN "0010110"
:SERIALBUS1:I2C:SEARCH:SETUP:GENERALCALL:BIT7MADDRESS:PATTERN?
-> :SERIALBUS1:I2C:SEARCH:SETUP:GENERALCALL:BIT7MADDRESS:PATTERN "0010110"
```

:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:SBYTe

Function Sets or queries the general call second byte type for I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:SBYTe {BIT7Maddress|DONTcare|H04|H06}
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:SBYTe?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:GENERALCALL:SBYTE BIT7MADDRESS
:SERIALBUS1:I2C:SEARCH:SETUP:GENERALCALL:SBYTE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:GENERALCALL:SBYTE BIT7MADDRESS
```

:SERialbus<x>:I2C:SEARch[:SETup]:MODE

Function Sets or queries the I2C bus signal search mode setting.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:MODE {ADRData|EVERystart|GENeralcall|HSMode|NONack|STARtbyte}
:SERialbus<x>:I2C:SEARch[:SETup]:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:MODE ADRDATA
:SERIALBUS1:I2C:SEARCH:SETUP:MODE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:MODE ADRDATA
```

:SERialbus<x>:I2C:SEARch[:SETup]:NONack?

Function Queries all I2C bus signal search NON ACK ignore mode settings.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:NONack?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:NONack:HSMode

Function Sets or queries whether NONACK is to be ignored in high-speed mode of I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:HSMode {<Boolean>}
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:HSMode?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:NONACK:HSMODE ON
:SERIALBUS1:I2C:SEARCH:SETUP:NONACK:HSMODE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:NONACK:HSMODE 1
```

:SERialbus<x>:I2C:SEARch[:SETup]:NONack:READaccess

Function Sets or queries whether NONACK is to be ignored in read access mode of I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:READaccess {<Boolean>}
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:READaccess?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:NONACK:READACCESS ON
:SERIALBUS1:I2C:SEARCH:SETUP:NONACK:READACCESS?
-> :SERIALBUS1:I2C:SEARCH:SETUP:NONACK:READACCESS 1
```

:SERialbus<x>:I2C:SEARch[:SETup]:NONAck:STARtbyte

Function Sets or queries whether NONACK is to be ignored in the start byte of I2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:STARtbyte {<Boolean>}
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:STARtbyte?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:NONACK:STARTBYTE ON
:SERIALBUS1:I2C:SEARCH:SETUP:NONACK:STARTBYTE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:NONACK:STARTBYTE 1
```

:SERialbus<x>:LIN?

Function Queries all LIN bus signal analysis and search settings.

Syntax
```
:SERialbus<x>:LIN?
```
<x> = 1 to 4

:SERialbus<x>:LIN:ANALyze?

Function Queries all LIN bus signal analysis settings.

Syntax
```
:SERialbus<x>:LIN:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:LIN[:ANALyze]:SETup?

Function Queries all LIN bus signal analysis bus settings.

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:LIN[:ANALyze]:SETup:BRATe

Function Sets or queries the LIN bus signal analysis bit rate (data transfer rate).

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup:BRATe {<NRf>|USER,<NRf>}
:SERialbus<x>:LIN[:ANALyze]:SETup:BRATe?
```
<x> = 1 to 4
<NRf> = 1200, 2400, 4800, 9600, 19200
USER <NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:BRATE 2400
:SERIALBUS1:LIN:ANALYZE:SETUP:BRATE?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:BRATE 2400
```

:SERialbus<x>:LIN[:ANALyze]:SETup:REVision

Function Sets or queries the LIN bus signal analysis revision number.

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup:REVision {BOTH|LIN1_3|LIN2_0}
:SERialbus<x>:LIN[:ANALyze]:SETup:REVision?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:REVISION LIN1_3
:SERIALBUS1:LIN:ANALYZE:SETUP:REVISION?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:REVISION LIN1_3
```

:SERialbus<x>:LIN[:ANALyze]:SETup:SOURce

Function Sets or queries the LIN bus signal analysis source.

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup:SOURce {<NRf>|MATH<y>}
:SERialbus<x>:LIN[:ANALyze]:SETup:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:SOURCE 1
:SERIALBUS1:LIN:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:SOURCE 1
```

:SERialbus<x>:LIN[:ANALyze]:SETup:SPOint

Function Sets or queries the LIN bus signal analysis sample point.

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup:SPOint {<NRf>}
:SERialbus<x>:LIN[:ANALyze]:SETup:SPOint?
```
<x> = 1 to 4
<NRf> = 18.8 to 90.6

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:SPOINT 18.8
:SERIALBUS1:LIN:ANALYZE:SETUP:SPOINT?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:SPOINT 18.8
```

:SERialbus<x>:LIN:DETail?

Function Queries all LIN bus signal analysis result list settings.

Syntax
```
:SERialbus<x>:LIN:DETail?
```
<x> = 1 to 4

:SERialbus<x>:LIN:DETail:DISPlay

Function Sets or queries the display mode for the LIN bus signal analysis result list.

Syntax
```
:SERialbus<x>:LIN:DETail:DISPl
ay {FULL|LOWer|UPPer}
:SERialbus<x>:LIN:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:DETAIL:DISPLAY FULL
:SERIALBUS1:LIN:DETAIL:DISPLAY?
-> :SERIALBUS1:LIN:DETAIL:DISPLAY FU
LL
```

:SERialbus<x>:LIN:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the LIN bus signal analysis result list.

Syntax
```
:SERialbus<x>:LIN:DETail:LIST:ITEM?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:DETAIL:LIST:ITEM?
-> :SERIALBUS1:LIN:DETAIL:LI
ST:ITEM "No.,Time(ms),ID,ID-
Field,Data,Checksum,Information"
```

:SERialbus<x>:LIN:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the LIN bus signal analysis result list.

Syntax
```
:SERialbus<x>:LIN:DETail:LIST:VAL
ue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:LIN:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:LIN:DETAIL:LIST:VALU
E "0,-1,0016,30,F0,2D04,CE,,"
```

:SERialbus<x>:LIN:SEARch?

Function Queries all LIN bus signal search settings.

Syntax
```
:SERialbus<x>:LIN:SEARch?
```
<x> = 1 to 4

:SERialbus<x>:LIN:SEARch:ABORt

Function Aborts the LIN bus signal search.

Syntax
```
:SERialbus<x>:LIN:SEARch:ABORt
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:ABORT
```

:SERialbus<x>:LIN:SEARch:EXECute

Function Executes a LIN bus signal search.

Syntax
```
:SERialbus<x>:LIN:SEARch:EXECute
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:EXECUTE
```

:SERialbus<x>:LIN:SEARch:FJUMp:BREak

Function Jumps to the break field in the LIN bus signal search result.

Syntax
```
:SERialbus<x>:LIN:SEARch:FJUMp:BREak
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:FJUMP:BREAK
```

:SERialbus<x>:LIN:SEARch:FJUMp:CSUM

Function Jumps to the checksum field in the LIN bus signal search result.

Syntax
```
:SERialbus<x>:LIN:SEARch:FJUMp:CSUM
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:FJUMP:CSUM
```

:SERialbus<x>:LIN:SEARch:FJUMp:DATA

Function Jumps to the data field in the LIN bus signal search result.

Syntax
```
:SERialbus<x>:LIN:SEARch:FJUMp:DATA
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:FJUMP:DATA
```

:SERialbus<x>:LIN:SEARch:FJUMp:IDENtifier

Function Jumps to the identifier field in the LIN bus signal search result.

Syntax
```
:SERialbus<x>:LIN:SEARch:FJUMp:IDENt
ifier
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:FJUMP:IDENTIF
IER
```

:SERialbus<x>:LIN:SEARch:FJUMp:SYNCh

Function Jumps to the sync field in the LIN bus signal search result.

Syntax
```
:SERialbus<x>:LIN:SEARch:FJUMp:SYNCh
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:FJUMP:SYNCH
```

:SERialbus<x>:LIN:SEARch:SELect

Function Sets which detected point to display in the LIN bus signal search zoom window and queries the zoom position of the detected point.

Syntax
```
:SERialbus<x>:LIN:SEARch:SELect {<NR
f>|MAXimum}
:SERialbus<x>:LIN:SEARch:SELect?
```
<x> = 1 to 4
<NRf> = 0 to 49999

Example
```
:SERIALBUS1:LIN:SEARCH:SELECT 1
:SERIALBUS1:LIN:SEARCH:SELECT?
-> :SERIALBUS1:LIN:SEARCH:SELECT 1.5
0000000
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:LIN:SEARch:SELect? MAXimum

Function Queries the number of detected points in the LIN bus signal search.

Syntax
```
:SERialbus<x>:LIN:SEARch:SELect? {MAXimum}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SELECT? MAXIMUM
-> :SERIALBUS1:LIN:SEARCH:SELECT 100
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:LIN:SEARch:SETup?

Function Queries all LIN bus signal search condition settings.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup?
```
<x> = 1 to 4

:SERialbus<x>:LIN:SEARch:SETup:ERRor?

Function Queries all LIN bus signal search error settings.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:ERRor?
```
<x> = 1 to 4

:SERialbus<x>:LIN:SEARch:SETup:ERRor:CHECksum

Function Sets or queries the LIN bus signal search checksum error setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:ERRor:CHECksum {<Boolean>}
:SERialbus<x>:LIN:SEARch:SETup:ERRor:CHECksum?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:CHECKSUM ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:CHECKSUM?
-> :SERIALBUS1:LIN:SEARCH:SETUP:ERROR:CHECKSUM 1
```

:SERialbus<x>:LIN:SEARch:SETup:ERRor:FRAMing

Function Sets or queries the LIN bus signal search framing error setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:ERRor:FRAMing {<Boolean>}
:SERialbus<x>:LIN:SEARch:SETup:ERRor:FRAMing?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:FRAMING ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:FRAMING?
-> :SERIALBUS1:LIN:SEARCH:SETUP:ERROR:FRAMING 1
```

:SERialbus<x>:LIN:SEARch:SETup:ERRor:PARity

Function Sets or queries the LIN bus signal search parity error setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:ERRor:PARity {<Boolean>}
:SERialbus<x>:LIN:SEARch:SETup:ERRor:PARity?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:PARITY ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:PARITY?
-> :SERIALBUS1:LIN:SEARCH:SETUP:ERROR:PARITY 1
```

:SERialbus<x>:LIN:SEARch:SETup:ERRor:SYNCh

Function Sets or queries the LIN bus signal search synch error setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:ERRor:SYNCh {<Boolean>}
:SERialbus<x>:LIN:SEARch:SETup:ERRor:SYNCh?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:SYNCH ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:SYNCH?
-> :SERIALBUS1:LIN:SEARCH:SETUP:ERROR:SYNCH 1
```

:SERialbus<x>:LIN:SEARch:SETup:ERRor:TIMeout

Function Sets or queries the LIN bus signal search timeout error setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:ERRor:TIMeout {<Boolean>}
:SERialbus<x>:LIN:SEARch:SETup:ERRor:TIMeout?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:TIMEOUT ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:TIMEOUT?
-> :SERIALBUS1:LIN:SEARCH:SETUP:ERROR:TIMEOUT 1
```

:SERialbus<x>:LIN:SEARch:SETup:IDData?

Function Queries all ID and data condition settings for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData?
```
<x> = 1 to 4

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA?

Function Queries all LIN bus signal search data settings.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA?
```
<x> = 1 to 4

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:CONDition

Function Sets or queries the LIN bus signal search data condition setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:CONDITION EQUAL
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DBYTe

Function Sets or queries the number of data bytes for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DBYTe {<NRf>}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DBYTe?
```
<x> = 1 to 4
<NRf> = 1 to 8

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DBYTE 1
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DBYTE?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DBYTE 1
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DECimal<y>

Function Sets or queries the data for LIN bus signal searching in decimal notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DECimal<y> {<NRf>}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DECIMAL1?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DECIMAL1 1
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DPATtern:HEXa

Function Sets the data for LIN bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DPATtern:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:HEXA "12"
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DPATtern:PATTern

Function Sets or queries the data for LIN bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DPATtern:PATTern {<String>}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DPATtern:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:ENDian

Function Sets or queries the data endian for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:ENDian {BIG|LITTle}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:ENDian?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:ENDIAN?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:ENDIAN BIG
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MODE

Function Sets or queries the data condition enable/disable state for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MODE {<Boolean>}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:MODE ON
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:MODE?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:MODE 1
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MSBLsb

Function Sets or queries the MSB/LSB bit for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MSBLsb?
```
<x> = 1 to 4
<NRf> = 0 to 63

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:MSBLSB 1,0
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:MSBLSB?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:MSBLSB 1,0
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:SIGN

Function Sets or queries the data sign for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:SIGN {SIGN|UNSign}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:SIGN?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:SIGN?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:SIGN SIGN
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier?

Function Queries all LIN bus signal search identifier settings.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier?
```
<x> = 1 to 4

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID?

Function Queries all LIN bus signal search ID settings.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID?
```
<x> = 1 to 4

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:HEXa

Function Sets the ID for LIN bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:IDENTIFIER:ID:HEXA "1E"
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:MODE

Function Sets or queries the ID condition enable/disable state for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:MODE {<Boolean>}
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:IDENTIFIER:ID:MODE ON
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:IDENTIFIER:ID:MODE?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:IDENTIFIER:ID:MODE 1
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:PATTern

Function Sets or queries the ID for LIN bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:PATTern{<String>}
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:IDENTIFIER:ID:PATTERN "101100"
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:IDENTIFIER:ID:PATTERN?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:IDENTIFIER:ID:PATTERN "101100"
```

:SERialbus<x>:LIN:SEARch:SETup:MODE

Function Sets or queries the LIN bus signal search mode setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:MODE {BSYNch|ERRor|IDData}
:SERialbus<x>:LIN:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:MODE BSYNCH
:SERIALBUS1:LIN:SEARCH:SETUP:MODE?
-> :SERIALBUS1:LIN:SEARCH:SETUP:MODE BSYNCH
```

:SERialbus<x>:RWINdow

Function Sets or queries the zoom window for showing the detected area.

Syntax
```
:SERialbus<x>:RWINdow {Z1|Z2}
:SERialbus<x>:RWINdow?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:RWINDOW Z1
:SERIALBUS1:RWINDOW?
-> :SERIALBUS1:RWINDOW Z1
```

:SERialbus<x>:SENT?

Function Queries all SENT signal analysis and search settings.

Syntax
```
:SERialbus<x>:SENT?
```
<x> = 1 to 4

:SERialbus<x>:SENT:ANALyze?

Function Queries all SENT signal analysis settings.

Syntax
```
:SERialbus<x>:SENT:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup?

Function Queries all bus setup settings for SENT signal analysis.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup:DISPlay

Function Sets or queries the display mode for the SENT signal analysis result.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:DISPlay {BOTH|FAST|SLOW}
:SERialbus<x>:SENT[:ANALyze]:SETup:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:DISPLAY BOTH
:SERIALBUS1:SENT:ANALYZE:SETUP:DISPLAY?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:DISPLAY BOTH
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST?

Function Queries all SENT signal analysis fast channel settings.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST?
```
<x> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:DTYPe

Function Sets or queries the SENT signal analysis fast channel data format.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:DTYPe {NIBBle|USER}
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:DTYPe?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:DTYPE NIBBLE
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:DTYPE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FAST:DTYPE NIBBLE
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup?

Function Queries all settings related to user-defined data of the SENT signal analysis fast channel.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup?
```
<x> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>?

Function Queries all settings related to user-defined data of the SENT signal analysis fast channel.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>?
```
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:MODE

Function Sets or queries the enable/disable condition of the specified user-defined data of the SENT signal analysis fast channel.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:MODE {<Boolean>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:MODE?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:MODE ON
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:MODE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:MODE 1
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:ORDer

Function Sets or queries the byte order of the specified user-defined data of the SENT signal analysis fast channel.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:ORDer {BIG|LITTle}
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:ORDer?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:ORDER BIG
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:ORDER?
->:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:ORDER BIG
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:SIZE

Function Sets or queries the data size of the specified user-defined data of the SENT signal analysis fast channel.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:SIZE {<NRf>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:SIZE?
```
<x> = 1 to 4
<y> = 1 to 4
<NRf> = 0 to 24

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:SIZE 24
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:SIZE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:DATA1:SIZE 24
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:MULTiplexing

Function Sets or queries the multiplexing enable/disable condition of the specified user-defined data of the SENT signal analysis fast channel.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:MULTiplexing {<Boolean>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:MULTiplexing?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:MULTIPLEXING ON
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:MULTIPLEXING?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FAST:USETUP:MULTIPLEXING 1
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat?

Function Queries all SENT signal analysis format settings.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat?
```
<x> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor?

Function Queries all SENT signal analysis error factor settings.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor?
```
<x> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SAComm?

Function Queries all status and communication error factor settings of SENT signal analysis.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:SAComm?
```
<x> = 1 to 4

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SAComm:BIT<y>?

Function Queries all status and communication error factor bit settings of SENT signal analysis.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:SAComm:BIT<y>?
```
<x> = 1 to 4
<y> = 0 to 3

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SCPulses

Function Sets or queries the consecutive calibration pulse error factor of SENT signal analysis.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SCPulses {OFF|OPT2|POPTion}
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SCPulses?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CEFACTOR:SCPULSES OFF
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CEFACTOR:SCPULSES?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CEFACTOR:SCPULSES OFF
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CRCType

Function Sets or queries the SENT signal analysis CRC computation type.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CRCType {LEGacy|RECommended}
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CRCType?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CRCTYPE LEGACY
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CRCTYPE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CRCTYPE LEGACY
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTICk

Function Sets or queries the SENT signal analysis clock tick value.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTICk {<NRf>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTICk?
```
<x> = 1 to 4
<NRf> = 1.00µs to 100.00µs (resolution: 0.01µs)

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CTICK 0.000001
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CTICK?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CTICK 1.000000E-06
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTOLerance

Function Sets or queries the SENT signal analysis clock tolerance.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTOLerance {<NRf>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTOLerance?
```
<x> = 1 to 4
<NRf> = ±1.0% to ±30.0% (resolution: 0.1%)

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CTOLerance 25.0
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CTOLERANCE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:CTOLERANCE 25.0E+00
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:DNIBbles

Function Sets or queries the number of data nibbles for SENT signal analysis.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:DNIBbles {<NRf>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:DNIBbles?
```
<x> = 1 to 4
<NRf> = 1 to 6

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:DNIBBLES 6
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:DNIBBLES?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:DNIBBLES 6
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:PPULse

Function Sets or queries the presence or absence of pause pulses for SENT signal analysis.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:PPULse {<Boolean>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:PPULse?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:PPULSE ON
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:PPULSE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:PPULSE 1
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:VERSion

Function Sets or queries the SENT signal analysis specification version.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:VERSion {APR2016|FEB2008|JAN2010}
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:VERESion?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:VERSION JAN2010
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:VERSION?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT:VERSION JAN2010
```

:SERialbus<x>:SENT[:ANALyze]:SETup:STYPe

Function Sets or queries the SENT signal analysis slow channel format.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:STYPe {ENHanced|SHORt}
:SERialbus<x>:SENT[:ANALyze]:SETup:STYPe?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:STYPE ENHANCED
:SERIALBUS1:SENT:ANALYZE:SETUP:STYPE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:STYPE ENHANCED
```

:SERialbus<x>:SENT[:ANALyze]:SETup:SOURce

Function Sets or queries the SENT signal analysis source.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:SOURce {<NRf>|BIT<y>|MATH<y>}
:SERialbus<x>:SENT[:ANALyze]:SETup:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 8
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:SOURCE 1
:SERIALBUS1:SENT:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:SOURCE 1
```

:SERialbus<x>:SENT:DETail?

Function Queries all SENT signal analysis list settings.

Syntax
```
:SERialbus<x>:SENT:DETail?
```
<x> = 1 to 4

:SERialbus<x>:SENT:DETail:DISPlay

Function Sets or queries the display mode for the SENT signal analysis list.

Syntax
```
:SERialbus<x>:SENT:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:SENT:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:DETAIL:DISPLAY FULL
:SERIALBUS1:SENT:DETAIL:DISPLAY?
-> :SERIALBUS1:SENT:DETAIL:DISPLAY FULL
```

:SERialbus<x>:SENT:DETail:LIST:ALL?

Function Queries all the data for all analysis numbers in the SENT signal analysis result list.

Syntax `:SERialbus<x>:SENT:DETail:LIST:ALL?`
<x> = 1 to 4

Example `:SERIALBUS1:SENT:DETAIL:LIST:ALL?`
`-> #8 (8-digit number of bytes)(data sequence)`

Description All the data for all analysis numbers are returned in <block data> format.
The data of each analysis number is separated by ASCII code 0AH.

:SERialbus<x>:SENT:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the SENT signal analysis result list.

Syntax `:SERialbus<x>:SENT:DETail:LIST:ITEM?`
<x> = 1 to 4

Example `:SERIALBUS1:SENT:DETail:LIST:ITEM?`
`-> :SERIALBUS1:SENT:DETAIL:LIST:ITEM "No.,Time(ms),Sync(us),Tick(us),S&C,Data,CRC,Length(tick),Information,SlowCH,"`

Description The items displayed in the analysis result list varies depending on the :SERialbus<x>:SENT[:ANALyze]:SETup:DISPlay {BOTH|FAST|SLOW} setting.

:SERialbus<x>:SENT:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the SENT signal analysis result list.

Syntax `:SERialbus<x>:SENT:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}`
<x> = 1 to 4
<NRf> = –100000 to 100000

Example `:SERIALBUS1:SENT:DETail:LIST:VALUE? 0`
`-> :SERIALBUS1:SENT:DETAIL:LIST:VALUE "0,-0.1672,168.00,3.00,1100,6,3,5,E,B,9,3,283.73,,,"`

Description The items displayed in the analysis result list varies depending on the :SERialbus<x>:SENT[:ANALyze]:SETup:DISPlay {BOTH|FAST|SLOW} setting.

:SERialbus<x>:SENT:SEARch?

Function Queries all SENT signal search settings.

Syntax `:SERialbus<x>:SENT:SEARch?`
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch:ABORt

Function Aborts the SENT signal search.

Syntax `:SERialbus<x>:SENT:SEARch:ABORt`
<x> = 1 to 4

Example `:SERIALBUS1:SENT:SEARCH:ABORT`

:SERialbus<x>:SENT:SEARch:EXECute

Function Executes a SENT signal search.

Syntax `:SERialbus<x>:SENT:SEARch:EXECute`
<x> = 1 to 4

Example `:SERIALBUS1:SENT:SEARCH:EXECUTE`

:SERialbus<x>:SENT:SEARch:SELect

Function Sets which detected point to display in the SENT signal search zoom window and queries the zoom position of the detected point.

Syntax `:SERialbus<x>:SENT:SEARch:SELect {<NRf>|MAXimum}`
`:SERialbus<x>:SENT:SEARch:SELect?`
<x> = 1 to 4
<NRf> = 0 to 49999

Example `:SERIALBUS1:SENT:SEARCH:SELECT 1`
`:SERIALBUS1:SENT:SEARCH:SELECT?`
`-> :SERIALBUS1:SENT:SEARCH:SELECT 1.50000000`

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:SENT:SEARch:SELect? MAXimum

Function Queries the number of detected points in the SENT signal search.

Syntax `:SERialbus<x>:SENT:SEARch:SELect? {MAXimum}`
<x> = 1 to 4

Example `:SERIALBUS1:SENT:SEARCH:SELECT? MAXimum`
`-> :SERIALBUS1:SENT:SEARCH:SELECT 100`

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:SENT:SEARch:SETup?

Function Queries all SENT signal search setup settings.

Syntax `:SERialbus<x>:SENT:SEARch:SETup?`
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:ERRor?

Function Queries all SENT signal search error settings.

Syntax `:SERialbus<x>:SENT:SEARch[:SETup]:ERRor?`
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:FCRC

Function Sets or queries the SENT signal search fast channel CRC error.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:FCRC {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:FCRC?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:FCRC ON
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:FCRC?
-> :SERIALBUS1:SENT:SEARCH:SETUP:ERROR:FCRC 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NDValue

Function Sets or queries the SENT signal search nibble data value error.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NDValue {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NDValue?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:NDVALUE ON
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:NDVALUE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:ERROR:NDVALUE 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NNUMber

Function Sets or queries the SENT signal search nibble data count error.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NNUMber {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NNUMber?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:NNUMBER ON
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:NNUMBER?
-> :SERIALBUS1:SENT:SEARCH:SETUP:ERROR:NNUMBER 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SAComm

Function Sets or queries the SENT signal search status and communication error.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SAComm {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SAComm?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SACOMM ON
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SACOMM?
-> :SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SACOMM 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCPulses

Function Sets or queries the consecutive calibration pulse error for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCPulses{<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCPulses?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SCPULSESON
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SCPULSES?
-> :SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SCPULSES 1
```

Description When :SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SCPulses is set to OFF, this setting is fixed to OFF.

:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCRC

Function Sets or queries the SENT signal search slow channel CRC error.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCRC {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCRC?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SCRC ON
:SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SCRC?
-> :SERIALBUS1:SENT:SEARCH:SETUP:ERROR:SCRC 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa?

Function Queries all fast channel data settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>?

Function Queries all settings related to the specified fast channel user data for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>?
```
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:CONDition

Function Sets or queries the comparison condition of the specified user data of the SENT signal search fast channel.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:CONDition?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:CONDITION EQUAL
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:CONDITION EQUAL
```

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:DECimal<z>

Function Sets or queries the data of the specified user data of the SENT signal search fast channel in decimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:DECimal<z> {<NRf>}
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:DECimal<z>?
```
<x> = 1 to 4
<y> = 1 to 4
<z> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:DECIMAL1 1
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:DECIMAL1 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:MODE

Function Sets or queries the data enable/disable condition of the specified user data of the SENT signal search fast channel.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:MODE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:MODE?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:MODE 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles?

Function Queries all fast channel nibble data settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:CONDition

Function Sets or queries the nibble data comparison condition of the SENT signal search fast channel.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:CONDition {FALSe|TRUE}
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:CONDITION TRUE
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:CONDITION TRUE
```

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:HEXa

Function Sets the SENT signal search fast channel nibble data in hexadecimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:HEXA "112233"
```

:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:PATTern

Function Sets or queries the SENT signal search fast channel nibble data in binary notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:PATTern {<String>}
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:PATTERN "111100001111000011110000"
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:PATTERN?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:PATTERN "111100001111000011110000"
```

:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm?

Function Queries all fast channel status and communication nibble settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:HEXa

Function Sets the SENT signal search fast channel status and communication nibble data in hexadecimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FSACOMM:HEXA "F"
```

:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:PATTern

Function Sets or queries the SENT signal search fast channel status and communication nibble data in binary notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:PATTern {<String>}
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FSACOMM:PATTERN "1111"
:SERIALBUS1:SENT:SEARCH:SETUP:FSACOMM:PATTERN?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FSACOMM:PATTERN "1111"
```

:SERialbus<x>:SENT:SEARch[:SETup]:MODE

Function Sets or queries the SENT signal search mode.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:MODE {EFASt|ERRor|ESLow|FDATa|FSAComm|SDATa}
:SERialbus<x>:SENT:SEARch[:SETup]:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:MODE ERRor
:SERIALBUS1:SENT:SEARCH:SETUP:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:MODE ERRor
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa?

Function Queries all slow channel data settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced?

Function Queries all slow channel enhanced type settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:CBIT

Function Sets or queries the SENT signal search slow channel enhanced type configuration bit.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:CBIT {D12Bit|D16Bit}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:CBIT?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:CBIT D12Bit
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:CBIT?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:CBIT D12Bit
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit?

Function Queries all slow channel enhanced type 12-bit data and 8-bit ID settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA?

Function Queries all slow channel enhanced type 12-bit data and 8-bit ID data settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:CONDition

Function Sets or queries the data comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:CONDition {BETWeen|EQUal|FALSe|GREater|LESS|NOTBetween|NOTEqual|TRUE}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:CONDITION FALSE
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:CONDITION FALSE
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:DECimal<y>

Function Sets or queries the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in decimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:DECimal<y> {<NRf>}
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:DECIMAL1 123
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:DECIMAL1 123
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:HEXa

Function Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:HEXA "123"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:MODE

Function Sets or queries the data enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:MODE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:MODE 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:PATTern

Function Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in binary notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:PATTern {<String>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:PATTERN "111101010101"
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:PATTERN?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:DATA:PATTERN "111101010101"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID?

Function Queries all ID settings related to the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:CONDition

Function Sets or queries the ID comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:CONDITION EQUAL
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:CONDITION EQUAL
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:DECimal<y>

Function Sets the ID of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching in decimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:DECimal<y> {<NRf>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:DECIMAL1 0
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:DECIMAL1 0
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:MODE

Function Sets or queries the ID enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:MODE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D12BIT:ID:MODE 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit?

Function Queries all slow channel enhanced type 16-bit data and 4-bit ID settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA?

Function Queries all slow channel enhanced type 16-bit data and 4-bit ID data settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:CONDition

Function Sets or queries the data comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:CONDition {BETWeen|EQUal|FALSe|GREater|LESS|NOTBetween|NOTEqual|TRUE}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D16BIT:DATA:CONDITION FALSE
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D16BIT:DATA:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:ENHANCED:D16BIT:DATA:CONDITION FALSE
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:DECimal<y>

Function Sets or queries the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in decimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:
SDATa:ENHanced:D16Bit:DATA:DECimal
<y> {<NRf>}
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:DATA:DECIMAL1 123
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:DATA:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDA
TA:ENHANCED:D16BIT:DATA:DECIMAL1 123
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:HEXa

Function Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup
]:SDATa:ENHanced:D16Bit:DATA:HE
Xa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:DATA:HEXA "0000"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:MODE

Function Sets or queries the data enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup
]:SDATa:ENHanced:D16Bit:DATA:MO
DE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:ENHanced:D16Bit:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:DATA:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:DATA:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDA
TA:ENHANCED:D16BIT:DATA:MODE 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:PATTern

Function Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in binary notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]
:SDATa:ENHanced:D16Bit:DATA:PATTe
rn {<String>}
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:ENHanced:D16Bit:DATA:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:S
DATA:ENHANCED:D16BIT:DATA:PATTE
RN "1111000011110000"
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:DATA:PATTERN?
-> :SERIALBUS1:SENT:SEARCH:SETUP:
SDATA:ENHANCED:D16BIT:DATA:PATTE
RN "1111000011110000"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:ID?

Function Queries all ID settings related to the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:ENHanced:D16Bit:ID?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:ID:CONDition

Function Sets or queries the ID comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:ENHanced:D16Bit:ID:CONDition {EQ
Ual|GREater|INSide|LESS|NOTEqual|OUT
Side}
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:ENHanced:D16Bit:ID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:ID:CONDITION EQUAL
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:ID:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:
SDATA:ENHANCED:D16BIT:ID:CONDITI
ON EQUAL
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:ID:DECimal<y>

Function Sets the ID of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching in decimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]
:SDATa:ENHanced:D16Bit:ID:DECimal
<y> {<NRf>}
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:ENHanced:D16Bit:ID:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:ID:DECIMAL1 0
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:ID:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDA
TA:ENHANCED:D16BIT:ID:DECIMAL1 0
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:ID:MODE

Function Sets or queries the ID enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETu
p]:SDATa:ENHanced:D16Bit:ID:MO
DE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:ENHanced:D16Bit:ID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:ID:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
ENHANCED:D16BIT:ID:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDA
TA:ENHANCED:D16BIT:ID:MODE 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt?

Function Queries all slow channel short type settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:SHORt?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA?

Function Queries all slow channel short type data settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:SHORt:DATA?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:CONDition

Function Sets or queries the short type data comparison condition of the SENT signal search slow channel.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:SHORt:DATA:CONDition {EQUal|FALS
E|GREater|INSide|LESS|NOTEqual|OUTSi
de|TRUE}
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:SHORt:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
SHORT:DATA:CONDITION EQUAL
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
SHORT:DATA:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDA
TA:SHORT:DATA:CONDITION EQUAL
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:DECimal<y>

Function Sets or queries the SENT signal search slow channel short type data in decimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:SHORt:DATA:DECimal<y> {<NRf>}
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
SHORt:DATA:DECIMAL1 123
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
SHORt:DATA:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDA
TA:SHORt:DATA:DECIMAL1 123
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:HEXa

Function Sets the SENT signal search slow channel short type data in hexadecimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SD
ATa:SHORt:DATA:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:
SHORT:DATA:HEXA "00"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:MODE

Function Sets or queries the short type data enable/disable condition of the SENT signal search slow channel.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:MODE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:MODE 1
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:PATTern

Function Sets the SENT signal search slow channel short type data in binary notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:PATTern {<String>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:PATTERN "11110000"
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:PATTERN?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:PATTERN "11110000"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID?

Function Queries all slow channel short type ID settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID?
```
<x> = 1 to 4

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:CONDition

Function Sets or queries the short type ID comparison condition of the SENT signal search slow channel.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:CONDITION EQUAL
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:CONDITION EQUAL
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:DECimal<y>

Function Sets the SENT signal search slow channel short type ID in decimal notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:DECimal<y> {<NRf>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:DECimal1 0
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:DECIMAL1 0
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:MODE

Function Sets or queries the short type ID enable/disable condition of the SENT signal search slow channel.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:MODE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:MODE 1
```

:SERialbus<x>:SENT:TRENd<y>?

Function Queries all SENT signal analysis trend display settings.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>?
```
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:SENT:TRENd<y>:ASCale

Function Executes auto scaling of the SENT signal analysis trend display.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:ASCale
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:ASCALE
```

:SERialbus<x>:SENT:TRENd<y>:CURSor?

Function Queries all SENT signal analysis trend display cursor measurement settings.

Syntax :SERialbus<x>:SENT:TRENd<y>:CURSor?
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:SENT:TRENd<y>:CURSor:DT:VALue?

Function Queries the time value between cursors on the SENT signal analysis trend display.

Syntax :SERialbus<x>:SENT:TRENd<y>:CURSor:DT:VALue?
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:SENT:TRENd<y>:CURSor:DV:VALue?

Function Queries the vertical value between cursors on the SENT signal analysis trend display.

Syntax :SERialbus<x>:SENT:TRENd<y>:CURSor:DV:VALue?
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:SENT:TRENd<y>:CURSor:MODE

Function Sets or queries the automated measurement mode of the SENT signal analysis trend display.

Syntax :SERialbus<x>:SENT:TRENd<y>:CURSor:MODE {<Boolean>}
:SERialbus<x>:SENT:TRENd<y>:CURSor:MODE?
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:CURSOR:MODE ON
:SERIALBUS1:SENT:TREND1:CURSOR:MODE?
-> :SERIALBUS1:SENT:TREND1:CURSOR:MODE 1
```

:SERialbus<x>:SENT:TRENd<y>:CURSor:POSition<z>

Function Sets or queries the position of the specified cursor on the SENT signal analysis trend display.

Syntax :SERialbus<x>:SENT:TRENd<y>:CURSor:POSition<z> {<NRf>}
:SERialbus<x>:SENT:TRENd<y>:CURSor:POSition<z>?
<x> = 1 to 4
<y> = 1 to 4
<z> = 1 or 2
<NRf> = –5 div to 5 div (in steps of 10 divisions/display record length)

Example
```
:SERIALBUS1:SENT:TREND1:CURSOR:POSITION1 2
:SERIALBUS1:SENT:TREND1:CURSOR:POSITION1?
-> :SERIALBUS1:SENT:TREND1:CURSOR:POSITION1 2.00E+00
```

:SERialbus<x>:SENT:TRENd<y>:CURSor:T<z>:VALue?

Function Queries the time value at the specified cursor on the SENT signal analysis trend display.

Syntax :SERialbus<x>:SENT:TRENd<y>:CURSor:T<z>:VALue?
<x> = 1 to 4
<y> = 1 to 4
<z> = 1 or 2

:SERialbus<x>:SENT:TRENd<y>:CURSor:V<z>:VALue?

Function Queries the vertical value at the specified cursor on the SENT signal analysis trend display.

Syntax :SERialbus<x>:SENT:TRENd<y>:CURSor:V<z>:VALue?
<x> = 1 to 4
<y> = 1 to 4
<z> = 1 or 2

:SERialbus<x>:SENT:TRENd<y>:DISPlay

Function Sets or queries whether to show (ON) or hide (OFF) the SENT signal analysis trend.

Syntax :SERialbus<x>:SENT:TRENd<y>:DISPlay {<Boolean>}
:SERialbus<x>:SENT:TRENd<y>:DISPlay?
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:DISPLAY ON
:SERIALBUS1:SENT:TREND1:DISPLAY?
-> :SERIALBUS1:SENT:TREND1:DISPLAY 1
```

:SERialbus<x>:SENT:TRENd<y>:FCONtrol

Function Sets or queries the frame control value of the SENT signal analysis trend display.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:FCONtr
ol {<String>}
:SERialbus<x>:SENT:TRENd<y>:FCONtr
ol?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:FCONTROL "1"
:SERIALBUS1:SENT:TREND1:FCONTROL?
-> :SERIALBUS1:SENT:TREND1:FCONTR
OL 1
```

:SERialbus<x>:SENT:TRENd<y>:HRANge

Function Sets or queries the SENT signal analysis trend display source window.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:HRAN
ge {MAIN|Z1|Z2}
:SERialbus<x>:SENT:TRENd<y>:HRANge?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:HRANGE MAIN
:SERIALBUS1:SENT:TREND1:HRANGE?
-> :SERIALBUS1:SENT:TREND1:HRANGE MA
IN
```

:SERialbus<x>:SENT:TRENd<y>:SID

Function Sets or queries the slow channel ID of the SENT signal analysis trend display in hexadecimal notation.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:SID {<Stri
ng>}
:SERialbus<x>:SENT:TRENd<y>:SID?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:SID "01"
:SERIALBUS1:SENT:TREND1:SID?
-> :SERIALBUS1:SENT:TREND1:SID "01"
```

Description
- If the slow channel type is set to Short, the value of the first digit of this command's parameter will be set.
- Set this command's parameter using a 2-digit hexadecimal regardless of the slow channel type setting.

:SERialbus<x>:SENT:TRENd<y>:SOURce

Function Sets or queries the SENT signal analysis trend display source channel.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:SOUR
ce {FAST|SLOW}
:SERialbus<x>:SENT:TRENd<y>:SOURce?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:SOURCE FAST
:SERIALBUS1:SENT:TREND1:SOURCE?
-> :SERIALBUS1:SENT:TREND1:SOURCE FA
ST
```

:SERialbus<x>:SENT:TRENd<y>:UDATa

Function Sets or queries the fast channel user-defined data of the SENT signal analysis trend display.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:UDATa {<
NRf>}
:SERialbus<x>:SENT:TRENd<y>:UDATa?
```
<x> = 1 to 4
<y> = 1 to 4
<NRf> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:UDATA 1
:SERIALBUS1:SENT:TREND1:UDATA?
-> :SERIALBUS1:SENT:TREND1:UDATA 1
```

:SERialbus<x>:SENT:TRENd<y>:VERTical

Function Sets or queries the vertical range of the SENT signal analysis trend display.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:VERTic
al {<NRf>,<NRf>}
:SERialbus<x>:SENT:TRENd<y>:VERTic
al?
```
<x> = 1 to 4
<y> = 1 to 4
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:SERIALBUS1:SENT:TREND1:VERTIC
AL 1,-1
:SERIALBUS1:SENT:TREND1:VERTICAL?
-> :SERIALBUS1:SENT:TREND1:VERTIC
AL 1.0000000E+00,-1.0000000E+00
```

:SERialbus<x>:SENT:TRENd<y>:VTDisplay

Function Sets or queries the on/off status of the VT waveform display on the SENT signal analysis trend display.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:VTDispl
ay {<Boolean>}
:SERialbus<x>:SENT:TRENd<y>:VTDispl
ay?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:VTDISPLAY ON
:SERIALBUS1:SENT:TREND1:VTDISPLAY?
-> :SERIALBUS1:SENT:TREND1:VTDISPLA
Y 1
```

:SERialbus<x>:SOURce?

Function Queries all analysis and search settings.

Syntax
```
:SERialbus<x>:SOURce?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}?

Function Queries all source waveform settings.

Syntax
```
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}?
```
<x> = 1 to 4 (1 on 2-channel models)
<y> of CHANnel<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:HYSTeresis

Function Sets or queries the source waveform hysteresis.

Syntax
```
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:HYSTeresis {<NRf>}
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:HYSTeresis?
```
<x> = 1 to 4 (1 on 2-channel models)
<y> of CHANnel<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = See the Features Guide for this information.

:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:LEVel

Function Sets or queries the source waveform level.

Syntax
```
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:LEVel {<Voltage>,<Voltage>}
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:LEVel?
```
<x> = 1 to 4 (1 on 2-channel models)
<y> of CHANnel<y> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<Voltage> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SOURCE:CHANNEL1:LEVEL 0V
:SERIALBUS1:SOURCE:CHANNEL1:LEVEL?
-> :SERIALBUS1:SOURCE:CHANNEL1:LEVEL 0.000E+00
```

:SERialbus<x>:SPATtern?

Function Queries all user-defined bus signal analysis and search settings.

Syntax
```
:SERialbus<x>:SPATtern?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern:ANALyze?

Function Queries all user-defined bus signal analysis execution settings.

Syntax
```
:SERialbus<x>:SPATtern:ANALyze?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup?

Function Queries all user-defined bus signal analysis settings.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup:BRATe

Function Sets or queries the user-defined bus signal analysis bit rate.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:BRATe {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:BRATe?
```
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = 1000 to 50000000 (bps)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:BRATE 1000
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:BRATE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:BRATE 1000
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk?

Function Queries all user-defined bus signal analysis clock signal settings.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:MODE

Function Sets or queries the enable/disable condition of the user-defined bus signal analysis clock signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:MODE {<Boolean>}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:MODE?
```
<x> = 1 to 4 (1 on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:MODE ON
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:MODE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:MODE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :CLOCk:POLarity

Function Sets or queries the slope of the user-defined bus signal analysis clock signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:POLarity {FALL|RISE}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:POLarity?
```
<x> = 1 to 4 (1 on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:POLARITY FALL
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:POLARITY?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:POLARITY FALL
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :CLOCk:SOURce

Function Sets or queries the user-defined bus signal analysis clock signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:SOURce {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:SOURce?
```
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:SOURCE 1
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:SOURCE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:CLOCK:SOURCE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :CS?

Function Queries all user-defined bus signal analysis chip select signal settings.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup :CS:ACTive

Function Sets or queries the active state of the user-defined bus signal analysis chip select signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS:ACTive {HIGH|LOW}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS:ACTive?
```
<x> = 1 to 4 (1 on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CS:ACTIVE HIGH
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CS:ACTIVE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:CS:ACTIVE HIGH
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :CS:SOURce

Function Sets or queries the user-defined bus signal analysis chip select signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS:SOURce {<NRf>|NONE}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS:SOURce?
```
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = 1 to 4 (NRf is invalid on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CS:SOURCE 1
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CS:SOURCE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:CS:SOURCE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :DATA?

Function Queries all user-defined bus signal analysis data signal settings.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup :DATA:ACTive

Function Sets or queries the active state of the user-defined bus signal analysis data signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:ACTive {HIGH|LOW}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:ACTive?
```
<x> = 1 to 4 (1 on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DATA:ACTIVE HIGH
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DATA:ACTIVE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:DATA:ACTIVE HIGH
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :DATA:SOURce

Function Sets or queries the user-defined bus signal analysis data signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:SOURce {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:SOURce?
```
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DATA:SOURCE 1
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DATA:SOURCE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:DATA:SOURCE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :LATCh?

Function Queries all user-defined bus signal analysis latch signal settings.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh?
```
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup :LATCh:POLarity

Function Sets or queries the slope of the user-defined bus signal analysis latch signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:POLarity {FALL|RISE}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:POLarity?
```
<x> = 1 to 4 (1 on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LATCH:POLARITY FALL
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LATCH:POLARITY?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:LATCH:POLARITY FALL
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :LATCh:SOURce

Function Sets or queries the user-defined bus signal analysis latch signal.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:SOURce {<NRf>|NONE}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:SOURce?
```
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = 1 to 4 (NRf is invalid on 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LATCH:SOURCE 1
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LATCH:SOURCE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:LATCH:SOURCE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup :SPOint

Function Sets or queries the analysis start point of the user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETup:SPOint {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETup:SPOint?
```
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = –5 div to 5 div

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:SPOINT -5
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:SPOINT?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP:SPOINT -5.0000000E+00
```

:SERialbus<x>:SPATtern:SEARch?

Function Queries all user-defined bus signal search settings.

Syntax :SERialbus<x>:SPATtern:SEARch?
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern:SEARch:ABORt

Function Aborts the user-defined bus signal search.

Syntax :SERialbus<x>:SPATtern:SEARch:ABORt
<x> = 1 to 4 (1 on 2-channel models)

Example :SERIALBUS1:SPATTERN:SEARCH:ABORT

:SERialbus<x>:SPATtern:SEARch:EXECute

Function Executes a user-defined bus signal search.

Syntax :SERialbus<x>:SPATtern:SEARch:EXECute
<x> = 1 to 4 (1 on 2-channel models)

Example :SERIALBUS1:SPATTERN:SEARCH:EXECUTE

:SERialbus<x>:SPATtern:SEARch:SELect

Function Sets which detected point to display in the user-defined bus signal search zoom window and queries the zoom position of the detected point.

Syntax :SERialbus<x>:SPATtern:SEARch:SELect {<NRf>|MAXimum}
:SERialbus<x>:SPATtern:SEARch:SELect?
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = 0 to 49999

Example :SERIALBUS1:SPATTERN:SEARCH:SELECT 1
:SERIALBUS1:SPATTERN:SEARCH:SELECT?
-> :SERIALBUS1:SPATTERN:SEARCH:SELECT 1.50000000

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:SPATtern:SEARch:SELect? Maximum

Function Queries the number of detected points in the user-defined bus signal search.

Syntax :SERialbus<x>:SPATtern:SEARch:SELect? {MAXimum}
<x> = 1 to 4 (1 on 2-channel models)

Example :SERIALBUS1:SPATtern:SEARCH:SELECT? MAXIMUM
-> :SERIALBUS1:SPATtern:SEARCH:SELECT 100

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:SPATtern:SEARch:SETup?

Function Queries all User-defined bus signal search setup settings.

Syntax :SERialbus<x>:SPATtern:SEARch:SETup?
<x> = 1 to 4 (1 on 2-channel models)

:SERialbus<x>:SPATtern:SEARch:SETup:BITSize

Function Sets or queries the user-defined bus signal search bit length.

Syntax :SERialbus<x>:SPATtern:SEARch:SETup:BITSize {<NRf>}
:SERialbus<x>:SPATtern:SEARch:SETup:BITSize?
<x> = 1 to 4 (1 on 2-channel models)
<NRf> = 1 to 128

Example :SERIALBUS1:SPATTERN:SEARCH:SETUP:BITSIZE 1
:SERIALBUS1:SPATTERN:SEARCH:SETUP:BITSIZE?
-> :SERIALBUS1:SPATTERN:SEARCH:SETUP:BITSIZE 1

:SERialbus<x>:SPATtern:SEARch:SETup:HEXa

Function Sets or queries the data conditions of the user-defined bus signal search in hexadecimal notation.

Syntax :SERialbus<x>:SPATtern:SEARch:SETup:HEXa {<String>}
<x> = 1 to 4 (1 on 2-channel models)

Example :SERIALBUS1:SPATTERN:SEARCH:SETUP:HEXA "12"

:SERialbus<x>:SPATtern:SEARch:SETup:PATTern

Function Sets or queries the data conditions of the user-defined bus signal search in binary notation.

Syntax :SERialbus<x>:SPATtern:SEARch:SETup:PATTern {<String>}
:SERialbus<x>:SPATtern:SEARch:SETup:PATTern?
<x> = 1 to 4 (1 on 2-channel models)

Example :SERIALBUS1:SPATTERN:SEARCH:SETUP:PATTERN "00110101"
:SERIALBUS1:SPATTERN:SEARCH:SETUP:PATTERN?
-> :SERIALBUS1:SPATTERN:SEARCH:SETUP:PATTERN "00110101"

:SERialbus<x>:SPI?

Function Queries all SPI bus signal analysis and search settings.

Syntax :SERialbus<x>:SPI?
<x> = 1 to 4

:SERialbus<x>:SPI:ANALyze?

Function Queries all SPI bus signal analysis settings.

Syntax :SERialbus<x>:SPI:ANALyze?
<x> = 1 to 4

:SERialbus<x>:SPI[:ANALyze]:SETup?

Function Queries all SPI bus signal analysis bus settings.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:SPI[:ANALyze]:SETup:BITOrder

Function Sets or queries the bit order of the SPI bus signal analysis data.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:BITorder {MSBFirst|LSBFirst}
:SERialbus<x>:SPI[:ANALyze]:SETup:BITorder?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:BITORDER MSBFIRST
:SERIALBUS1:SPI:ANALYZE:SETUP:BITORDER?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:BITORDER MSBFIRST
```

:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk?

Function Queries all SPI bus signal analysis clock signal settings.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk?
```
<x> = 1 to 4

:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:POLarity

Function Sets or queries the slope of the SPI bus signal analysis clock signal.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:POLarity {FALL|RISE}
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:POLarity?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:CLOCK:POLARITY FALL
:SERIALBUS1:SPI:ANALYZE:SETUP:CLOCK:POLARITY?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:CLOCK:POLARITY FALL
```

:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:SOURce

Function Sets or queries the SPI bus signal analysis clock signal.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:SOURce {<NRf>|BIT<y>|MATH<y>}
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:CLOCK:SOURCE 1
:SERIALBUS1:SPI:ANALYZE:SETUP:CLOCK:SOURCE?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:CLOCK:SOURCE 1
```

:SERialbus<x>:SPI[:ANALyze]:SETup:CS?

Function Queries all SPI bus signal analysis chip select signal settings.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:CS?
```
<x> = 1 to 4

:SERialbus<x>:SPI[:ANALyze]:SETup:CS:ACTive

Function Sets or queries the active state of the SPI bus signal analysis chip select signal.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:CS:ACTive {HIGH|LOW}
:SERialbus<x>:SPI[:ANALyze]:SETup:CS:ACTive?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:CS:ACTIVE HIGH
:SERIALBUS1:SPI:ANALYZE:SETUP:CS:ACTIVE?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:CS:ACTIVE HIGH
```

:SERialbus<x>:SPI[:ANALyze]:SETup:CS:SOURce

Function Sets or queries the SPI bus signal analysis chip select signal.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:CS:SOURce {<NRf>|BIT<y>|MATH<y>}
:SERialbus<x>:SPI[:ANALyze]:SETup:CS:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:CS:SOURCE 1
:SERIALBUS1:SPI:ANALYZE:SETUP:CS:SOURCE?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:CS:SOURCE 1
```

:SERialbus<x>:SPI[:ANALyze]:SETup:DATA<y>?

Function Queries all SPI bus signal analysis data signal settings.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:DATA<y>?
```
<x> = 1 to 4
<y> = 1 or 2

:SERialbus<x>:SPI[:ANALyze]:SETup:DATA<y>:SOURce

Function Sets or queries the SPI bus signal analysis data signal.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:DATA<y>:SOURce {<NRf>|BIT<z>|MATH<z>}
:SERialbus<x>:SPI[:ANALyze]:SETup:DATA<y>:SOURce?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 to 4
<z> of BIT<z> = 1 to 8
<z> of MATH<z> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:DATA1:SOURCE 1
:SERIALBUS1:SPI:ANALYZE:SETUP:DATA1:SOURCE?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:DATA1:SOURCE 1
```

:SERialbus<x>:SPI[:ANALyze]:SETup:FIELd

Function Sets or queries the field size of the SPI bus signal analysis data.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:FIELd {<NRf>}
:SERialbus<x>:SPI[:ANALyze]:SETup:FIELd?
```
<x> = 1 to 4
<NRf> = 4 to 32

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:FIELD 4
:SERIALBUS1:SPI:ANALYZE:SETUP:FIELD?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:FIELD 4
```

SERialbus<x>:SPI[:ANALyze]:SETup:GROuping

Function Sets or queries the grouping on/off state for when the SPI bus signal analysis chip select signal is not available.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:GROuping {<Boolean>}
:SERialbus<x>:SPI[:ANALyze]:SETup:GROuping?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:GROUPING ON
:SERIALBUS1:SPI:ANALYZE:SETUP:GROUPING?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:GROUPING 1
```

:SERialbus<x>:SPI[:ANALyze]:SETup:ITIMe

Function Sets or queries the idle time for when the SPI bus signal analysis chip select signal is not available.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:ITIMe {<Time>}
:SERialbus<x>:SPI[:ANALyze]:SETup:ITIMe?
```
<x> = 1 to 4
<Time> = 10 ns to 1 ms

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:ITIME 2US
:SERIALBUS1:SPI:ANALYZE:SETUP:ITIME?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:ITIME 2.00000E-06
```

:SERialbus<x>:SPI[:ANALyze]:SETup:MODE

Function Sets or queries the wiring system (three-wire system or four-wire system) of the SPI bus signal analysis data signal.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:MODE {WIRE3|WIRE4}
:SERialbus<x>:SPI[:ANALyze]:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:MODE WIRE3
:SERIALBUS1:SPI:ANALYZE:SETUP:MODE?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:MODE WIRE3
```

:SERialbus<x>:SPI[:ANALyze]:SETup:MSBLsb

Function Sets or queries the MSB/LSB bit for SPI bus signal analysis data.

Syntax
```
:SERialbus<x>:SPI[:ANALyze]:SETup:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:SPI[:ANALyze]:SETup:MSBLsb?
```
<x> = 1 to 4
<NRf> = 0 to 31

Example
```
:SERIALBUS1:SPI:ANALYZE:SETUP:MSBLSB 7,0
:SERIALBUS1:SPI:ANALYZE:SETUP:MSBLSB?
-> :SERIALBUS1:SPI:ANALYZE:SETUP:MSBLSB 7,0
```

:SERialbus<x>:SPI:DETail?

Function Queries all SPI bus signal analysis result list settings.

Syntax
```
:SERialbus<x>:SPI:DETail?
```
<x> = 1 to 4

:SERialbus<x>:SPI:DETail:DISPlay

Function Sets or queries the display mode for the SPI bus signal analysis result list.

Syntax
```
:SERialbus<x>:SPI:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:SPI:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:DETAIL:DISPLAY FULL
:SERIALBUS1:SPI:DETAIL:DISPLAY?
-> :SERIALBUS1:SPI:DETAIL:DISPLAY FULL
```

:SERialbus<x>:SPI:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the SPI bus signal analysis result list.

Syntax
```
:SERialbus<x>:SPI:DETail:LIST:ITEM?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:DETAIL:LIST:ITEM?
-> :SERIALBUS1:SPI:DETAIL:LIST:ITEM "No.,Time(ms),Data,,"
```

:SERialbus<x>:SPI:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the SPI bus signal analysis result list.

Syntax
```
:SERialbus<x>:SPI:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4
<NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:SPI:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:SPI:DETAIL:LIST:VALUE "0,0.077002,Data1,00 00 00 00 00 00 00 00,"
```

:SERialbus<x>:SPI:SEARch?

Function Queries all SPI bus signal search settings.

Syntax
```
:SERialbus<x>:SPI:SEARch?
```
<x> = 1 to 4

:SERialbus<x>:SPI:SEARch:ABORt

Function Aborts the SPI bus signal search.

Syntax
```
:SERialbus<x>:SPI:SEARch:ABORt
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:ABORT
```

:SERialbus<x>:SPI:SEARch:EXECute

Function Executes an SPI bus signal search.

Syntax
```
:SERialbus<x>:SPI:SEARch:EXECute
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:EXECUTE
```

:SERialbus<x>:SPI:SEARch:SELect

Function Sets which detected point to display in the SPI bus signal search zoom window and queries the zoom position of the detected point.

Syntax
```
:SERialbus<x>:SPI:SEARch:SELect {<NRf>|MAXimum}
:SERialbus<x>:SPI:SEARch:SELect?
```
<x> = 1 to 4
<NRf> = 0 to 49999

Example
```
:SERIALBUS1:SPI:SEARCH:SELECT 1
:SERIALBUS1:SPI:SEARCH:SELECT?
-> :SERIALBUS1:SPI:SEARCH:SELECT 1.50000000
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:SPI:SEARch:SELect?MAXimum

Function Queries the number of detected points in the SPI bus signal search.

Syntax
```
:SERialbus<x>:SPI:SEARch:SELect? {MAXimum}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:SELECT? MAXIMUM
-> :SERIALBUS1:SPI:SEARCH:SELECT 100
```

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:SPI:SEARch:SETup?

Function Queries all SPI bus signal search setup settings.

Syntax
```
:SERialbus<x>:SPI:SEARch:SETup?
```
<x> = 1 to 4

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>?

Function Queries all settings related to the specified data for SPI bus signal searching.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>?
```
<x> = 1 to 4
<y> = 1 or 2

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:BCOunt

Function Sets or queries the pattern comparison start position of the specified data for SPI bus signal searching.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:BCOunt {<NRf>}
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:BCOunt?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 0 to 9999

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:BCOUNT 0
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:BCOUNT?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1:BCOUNT 0
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:CONDition

Function Sets or queries the judgment method (true/false) of the specified data for SPI bus signal searching.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:CONDition {FALSe|TRUE}
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:CONDition?
```
<x> = 1 to 4
<y> = 1 or 2

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:CONDITION DONTCARE
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:CONDITION?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1:CONDITION DONTCARE
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DBYTe

Function Sets or queries the data size (number of bytes) of the specified data for SPI bus signal searching.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DBYTe {<NRf>}
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DBYTe?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DBYTE 1
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DBYTE?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DBYTE 1
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DPATtern:HEXa

Function Sets the specified data for SPI bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DPATtern:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DPATTERN:HEXA "AB"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DBYTe setting.

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DPATtern:PATTern

Function Sets or queries the specified data for SPI bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DPATtern:PATTern {<String>}
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DPATtern:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DPATTERN:PATTERN "11001010"
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DPATTERN:PATTERN?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DPATTERN:PATTERN "11001010"
```

Description The number of bytes that can be set varies depending on the :SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DBYTe setting.

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:MODE

Function Sets or queries the data condition enable/disable state for SPI bus signal searching.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:MODE {<Boolean>}
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:MODE?
```
<x> = 1 to 4
<y> = 1 or 2

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:MODE ON
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:MODE?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1:MODE 1
```

:SERialbus<x>:TYPE

Function Sets or queries the search type.

Syntax
```
:SERialbus<x>:TYPE {CANBus|CANFDbus|CXPIbus|FLEXray|I2CBus|LINBus|PSI5|SENT|SPATtern|SPIBus|UART}
:SERialbus<x>:TYPE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:TYPE CANBUS
:SERIALBUS1:TYPE?
-> :SERIALBUS1:TYPE CANBUS
```

:SERialbus<x>:UART?

Function Queries all UART signal analysis and search settings.

Syntax
```
:SERialbus<x>:UART?
```
<x> = 1 to 4

:SERialbus<x>:UART:ANALyze?

Function Queries all UART signal analysis settings.

Syntax
```
:SERialbus<x>:UART:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:UART[:ANALyze]:SETup?

Function Queries all bus setup settings for UART signal analysis.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:UART[:ANALyze]:SETup:BITorder

Function Sets or queries the UART signal analysis bit order.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:BITorder {MSBFirst|LSBFirst}
:SERialbus<x>:UART[:ANALyze]:SETup:BITorder?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:ANALYZE:SETUP:BITORDER MSBFIRST
:SERIALBUS1:UART:ANALYZE:SETUP:BITORDER?
-> :SERIALBUS1:UART:ANALYZE:SETUP:BITORDER MSBFIRST
```

:SERialbus<x>:UART[:ANALyze]:SETup:BRATe

Function Sets or queries the UART signal analysis bit rate (data transfer rate).

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:BRATe {<NRf>|USER,<NRf>}
:SERialbus<x>:UART[:ANALyze]:SETup:BRATe?
```
<x> = 1 to 4
<NRf> = 1200, 2400, 4800, 9600, 19200
USER <NRf> = See the Features Guide for this information.

Example
```
:SERIALBUS1:UART:ANALYZE:SETUP:BRATE 2400
:SERIALBUS1:UART:ANALYZE:SETUP:BRATE?
-> :SERIALBUS1:UART:ANALYZE:SETUP:BRATE 2400
```

:SERialbus<x>:UART[:ANALyze]:SETup:BSPace

Function Sets or queries the UART signal analysis byte space.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:BSPace {<Time>}
:SERialbus<x>:UART[:ANALyze]:SETup:BSPace?
```
<x> = 1 to 4
<Time> = See the Features Guide for this information.

Example
```
:SERIALBUS1:UART:ANALYZE:SETUP:BSPACE 0.1S
:SERIALBUS1:UART:ANALYZE:SETUP:BSPACE?
-> :SERIALBUS1:UART:ANALYZE:SETUP:BSPACE 1.000E-01
```

:SERialbus<x>:UART[:ANALyze]:SETup:GROuping

Function Sets or queries on/off state of UART signal analysis grouping.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:GROuping {<Boolean>}
:SERialbus<x>:UART[:ANALyze]:SETup:GROuping?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:ANALYZE:SETUP:GROUPING ON
:SERIALBUS1:UART:ANALYZE:SETUP:GROUPING?
-> :SERIALBUS1:UART:ANALYZE:SETUP:GROUPING 1
```

:SERialbus<x>:UART[:ANALyze]:SETup:POLarity

Function Sets or queries the UART signal analysis polarity.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:POLarity {NEGative|POSitive}
:SERialbus<x>:UART[:ANALyze]:SETup:POLarity?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:ANALYZE:SETUP:POLARITY NEGATIVE
:SERIALBUS1:UART:ANALYZE:SETUP:POLARITY?
-> :SERIALBUS1:UART:ANALYZE:SETUP:POLARITY NEGATIVE
```

:SERialbus<x>:UART[:ANALyze]:SETup:SOURce

Function Sets or queries the UART signal analysis signal.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:SOURce {<NRf>|BIT<y>|MATH<y>}
:SERialbus<x>:UART[:ANALyze]:SETup:SOURce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 to 4

Example
```
:SERIALBUS1:UART:ANALYZE:SETUP:SOURCE 1
:SERIALBUS1:UART:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:UART:ANALYZE:SETUP:SOURCE 1
```

:SERialbus<x>:UART[:ANALyze]:SETup:SPOint

Function Sets or queries the UART signal analysis sample point.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:SPOint {<NRf>}
:SERialbus<x>:UART[:ANALyze]:SETup:SPOint?
```
<x> = 1 to 4
<NRf> = 18.8 to 90.6

Example
```
:SERIALBUS1:UART:ANALYZE:SETUP:SPOINT 18.8
:SERIALBUS1:UART:ANALYZE:SETUP:SPOINT?
-> :SERIALBUS1:UART:ANALYZE:SETUP:SPOINT 18.8
```

:SERialbus<x>:UART:DETail?

Function Queries all UART signal analysis result list settings.

Syntax :SERialbus<x>:UART:DETail?
<x> = 1 to 4

:SERialbus<x>:UART:DETail:DISPlay

Function Sets or queries the display mode for the UART signal analysis result list.

Syntax :SERialbus<x>:UART:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:UART:DETail:DISPlay?
<x> = 1 to 4

Example :SERIALBUS1:UART:DETAIL:DISPLAY FULL
:SERIALBUS1:UART:DETAIL:DISPLAY?
-> :SERIALBUS1:UART:DETAIL:DISPLAY FULL

:SERialbus<x>:UART:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the UART signal analysis result list.

Syntax :SERialbus<x>:UART:DETail:LIST:ITEM?
<x> = 1 to 4

Example :SERIALBUS1:UART:DETAIL:LIST:ITEM?
-> :SERIALBUS1:UART:DETAIL:LIST:ITEM "No.,Time(ms),Data(HEX),Data(ASCII),Information"

:SERialbus<x>:UART:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the UART signal analysis result list.

Syntax :SERialbus<x>:UART:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}
<x> = 1 to 4
<NRf> = See the Features Guide for this information.

Example :SERIALBUS1:UART:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:UART:DETAIL:LIST:VALUE "0,-0.5720,3A 74 69 D2,":ti.",,"

:SERialbus<x>:UART:SEARch?

Function Queries all UART signal search settings.

Syntax :SERialbus<x>:UART:SEARch?
<x> = 1 to 4

:SERialbus<x>:UART:SEARch:ABORt

Function Aborts the UART signal search.

Syntax :SERialbus<x>:UART:SEARch:ABORt
<x> = 1 to 4

Example :SERIALBUS1:UART:SEARCH:ABORT

:SERialbus<x>:UART:SEARch:EXECute

Function Executes a UART signal search.

Syntax :SERialbus<x>:UART:SEARch:EXECute
<x> = 1 to 4

Example :SERIALBUS1:UART:SEARCH:EXECUTE

:SERialbus<x>:UART:SEARch:SELect

Function Sets which detected point to display in the UART signal search zoom window and queries the zoom position of the detected point.

Syntax :SERialbus<x>:UART:SEARch:SELect {<NRf>|MAXimum}
:SERialbus<x>:UART:SEARch:SELect?
<x> = 1 to 4
<NRf> = 0 to 49999

Example :SERIALBUS1:UART:SEARCH:SELECT 1
:SERIALBUS1:UART:SEARCH:SELECT?
-> :SERIALBUS1:UART:SEARCH:SELECT 1.50000000

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:UART:SEARch:SELect? Maximum

Function Queries the number of detected points in the UART signal search.

Syntax :SERialbus<x>:UART:SEARch:SELect? {MAXimum}
<x> = 1 to 4

Example :SERIALBUS1:UART:SEARCH:SELECT? MAXIMUM
-> :SERIALBUS1:UART:SEARCH:SELECT 100

Description If there are no detected points, the instrument returns “NAN.”

:SERialbus<x>:UART:SEARch:SETup?

Function Queries all search conditions for UART signal searching.

Syntax :SERialbus<x>:UART:SEARch:SETup?
<x> = 1 to 4

:SERialbus<x>:UART:SEARch:SETup:DATA?

Function Queries all UART signal search data settings.

Syntax :SERialbus<x>:UART:SEARch:SETup:DATA?
<x> = 1 to 4

:SERialbus<x>:UART:SEARch:SETup:DATA:ASCii

Function Sets the UART signal search data in ASCII format.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA
:ASCii {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:A
SCII "TEST"
```

:SERialbus<x>:UART:SEARch:SETup:DATA:CONDition

Function Sets or queries the data judgment method (true/false) for UART signal searching.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA
:CONDition {DONTcare|TRUE}
:SERialbus<x>:UART:SEARch:SETup:DATA
:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:C
ONDITION DONTCARE
:SERIALBUS1:UART:SEARCH:SETUP:DATA:C
ONDITION?
-> :SERIALBUS1:UART:SEARCH:SETUP:DAT
A:CONDITION DONTCARE
```

:SERialbus<x>:UART:SEARch:SETup:DATA:CSENsitive

Function Sets or queries whether to distinguish uppercase and lowercase letters in ASCII data for the UART signal search.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA
:CSENsitive {<Boolean>}
:SERialbus<x>:UART:SEARch:SETup:DATA
:CSENsitive?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:C
SENSITIVE ON
:SERIALBUS1:UART:SEARCH:SETUP:DATA:C
SENSITIVE?
-> :SERIALBUS1:UART:SEARCH:SETUP:DAT
A:CSENSITIVE 1
```

:SERialbus<x>:UART:SEARch:SETup:DATA:DBYTe

Function Sets or queries the number of data bytes for UART signal searching.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA
:DBYTe {<NRf>}
:SERialbus<x>:UART:SEARch:SETup:DATA
:DBYTe?
```
<x> = 1 to 4
<NRf> = 1 to 4 (byte)

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:D
BYTE 1
:SERIALBUS1:UART:SEARCH:SETUP:DATA:D
BYTE?
-> :SERIALBUS1:UART:SEARCH:SETUP:DAT
A:DBYTE 1
```

:SERialbus<x>:UART:SEARch:SETup:DATA:DPATtern:HEXa

Function Sets the data for UART signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA
:DPATtern:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:D
PATTERN:HEXA "12"
```

:SERialbus<x>:UART:SEARch:SETup:DATA:DPATtern:PATTern

Function Sets or queries the data for UART signal searching in binary notation.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA
:DPATtern:PATTern {<String>}
:SERialbus<x>:UART:SEARch:SETup:DATA
:DPATtern:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:D
PATTERN:PATTERN "00110101"
:SERIALBUS1:UART:SEARCH:SETUP:DATA:D
PATTERN:PATTERN?
-> :SERIALBUS1:UART:SEARCH:SETUP:DAT
A:DPATTERN:PATTERN "00110101"
```

:SERialbus<x>:UART:SEARch:SETup:DATA:PFORmat

Function Sets or queries the data input format, which is one of the ID and data conditions, for UART signal searching.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA:PFORmat {ASCii|PATTern}
:SERialbus<x>:UART:SEARch:SETup:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:PFORMAT PATTERN
:SERIALBUS1:UART:SEARCH:SETUP:DATA:PFORMAT?
-> :SERIALBUS1:UART:SEARCH:SETUP:DATA:PFORMAT PATTERN
```

:SERialbus<x>:UART:SEARch:SETup:ERRor?

Function Queries all UART signal search error settings.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:ERRor?
```
<x> = 1 to 4

:SERialbus<x>:UART:SEARch:SETup:ERRor:FRAMing

Function Sets or queries the UART signal search framing error setting.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:ERRor:FRAMing {<Boolean>}
:SERialbus<x>:UART:SEARch:SETup:ERRor:FRAMing?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:FRAMING ON
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:FRAMING?
-> :SERIALBUS1:UART:SEARCH:SETUP:ERROR:FRAMING 1
```

:SERialbus<x>:UART:SEARch:SETup:ERRor:PARity

Function Sets or queries the UART signal search parity error setting.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:ERRor:PARity {<Boolean>}
:SERialbus<x>:UART:SEARch:SETup:ERRor:PARity?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:PARITY ON
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:PARITY?
-> :SERIALBUS1:UART:SEARCH:SETUP:ERROR:PARITY 1
```

:SERialbus<x>:UART:SEARch:SETup:ERRor:PMODe

Function Sets or queries the UART signal search parity mode setting.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:ERRor:PMODe {EVEN|ODD}
:SERialbus<x>:UART:SEARch:SETup:ERRor:PMODe?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:PMODE EVEN
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:PMODE?
-> :SERIALBUS1:UART:SEARCH:SETUP:ERROR:PMODE EVEN
```

:SERialbus<x>:UART:SEARch:SETup:FORMat

Function Sets or queries the UART signal search format setting.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:FORMat {BIT7Parity|BIT8Parity|BIT8Noparity}
:SERialbus<x>:UART:SEARch:SETup:FORMat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:FORMAT BIT7PARITY
:SERIALBUS1:UART:SEARCH:SETUP:FORMAT?
-> :SERIALBUS1:UART:SEARCH:SETUP:FORMAT BIT7PARITY
```

:SERialbus<x>:UART:SEARch:SETup:MODE

Function Sets or queries the UART signal search mode setting.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:MODE {DATA|EDATA|ERRor}
:SERialbus<x>:UART:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:MODE DATA
:SERIALBUS1:UART:SEARCH:SETUP:MODE?
-> :SERIALBUS1:UART:SEARCH:SETUP:MODE DATA
```

:SERialbus<x>:ZLINkage

Function Sets or queries the linkage (on/off) between the result numbers and zoom positions of the serial bus signal analysis results.

Syntax
```
:SERialbus<x>:ZLINkage {<Boolean>}
:SERialbus<x>:ZLINkage?
```
<x> = 1 to 4 (1 on 2-channel models)

Example
```
:SERIALBUS1:ZLINKAGE ON
:SERIALBUS1:ZLINKAGE?
-> :SERIALBUS1:ZLINKAGE 1
```

5.25 SNAP Group

:SNAP

Function Takes a snapshot.

Syntax :SNAP

Example :SNAP

5.26 SSTart Group

:SSTart?

Function Sets the trigger mode to single, starts waveform acquisition. The instrument returns 0 if the waveform acquisition is stopped within the specified timeout. Otherwise, the instrument returns 1.

Syntax `:SSTart? {<NRf>}`
<NRf> = 1 to 36000 (timeout value in 100 ms steps; start and wait)
<NRf> = 0 (start only, no timeout)
<NRf> = –36000 to 1 (timeout value in 100 ms steps; wait without starting)

Example
```
:SSTART?
-> :SSTART 1
```

Description
- If the specified timeout is a positive value, the instrument starts waveform acquisition in single trigger mode and waits for the acquisition to be stopped within the specified timeout period.
- If the specified timeout is 0, the instrument starts waveform acquisition and returns 0 without waiting.
- If the specified timeout is a negative value, the instrument does not start waveform acquisition and simply waits for the acquisition to be stopped within the specified timeout period.

5.27 STARt Group

:STARt

Function Starts waveform acquisition.

Syntax :STARt

Example :START

Description Stop the waveform acquisition using “STOP”.

5.28 STATus Group

The commands in this group are used to make settings and queries related to the communication status feature. There are no front panel keys that correspond to the commands in this group. For information about status reports, see chapter 6.

:STATus?

Function Queries all the settings of the communication status feature.

Syntax `:STATus?`

:STATus:CONDition?

Function Queries the contents of the condition register.

Syntax `:STATus:CONDition?`

Example
```
:STATUS:CONDITION?
-> :STATUS:CONDITION 16
```

Description For information about the condition register, see chapter 6, “Status Reports.”

:STATus:EESE

Function Sets or queries the extended event enable register.

Syntax
```
:STATus:EESE {<Register>}
:STATus:EESE?
```
<Register> = 0 to 65535

Example
```
:STATUS:EESE 257
:STATUS:EESE?
-> :STATUS:EESE 257
```

Description For information about the extended event enable register, see chapter 6, “Status Reports.”

:STATus:EESR?

Function Queries the contents of the extended event register and clears the register.

Syntax `:STATus:EESR?`

Example
```
:STATUS:EESR?
-> :STATUS:EESR 1
```

Description For information about the extended event register, see chapter 6, “Status Reports.”

:STATus:ERRor?

Function Queries the error code and message of the last error that has occurred (top of the error queue).

Syntax `:STATus:ERRor?`

Example
```
:STATUS:ERROR?
-> 113,"Undefined header"
```

:STATus:FILTer<x>

Function Sets or queries transition filter.

Syntax
```
:STATus:FILTer<x> {BOTH|FALL|NEVer|RISE}
:STATus:FILTer<x>?
```
<x> = 1 to 16

Example
```
:STATUS:FILTER2 RISE
:STATUS:FILTER2?
-> :STATUS:FILTER2 RISE
```

Description For information about the transition filter, see chapter 6, “Status Reports.”

:STATus:QENable

Function Sets or queries whether messages other than errors will be stored to the error queue.

Syntax
```
:STATus:QENable {<Boolean>}
:STATus:QENable?
```

Example
```
:STATUS:QENABLE ON
:STATUS:QENABLE?
-> :STATUS:QENABLE 1
```

:STATus:QMESsage

Function Sets or queries whether message information will be attached to the response to the STATus:ERRor? query.

Syntax
```
:STATus:QMESsage {<Boolean>}
:STATus:QMESsage?
```

Example
```
:STATUS:QMESSAGE OFF
:STATUS:QMESSAGE?
-> :STATUS:QMESSAGE 0
```

:STATus:SPOLl? (Serial Poll)

Function Executes serial polling.

Syntax `:STATus:SPOLl?`

Example
```
:STATUS:SPOLL?
-> :STATUS:SPOLL 0
```

5.29 STOP Group

:STOP

Function Stops waveform acquisition.

Syntax `:STOP`

Example `:STOP`

Description Start the waveform acquisition using “STARt”.

5.30 STORe Group

:STORe?

Function Queries all the information related to setup data in the internal memory.

Syntax `:STORe?`

:STORe:SETup<x>?

Function Queries information about the setup data in the specified location of the internal memory.

Syntax `:STORe:SETup<x>?`
<x> = 1 to 3

:STORe:SETup<x>:COMMent

Function Sets or queries the comment for the setup data that is stored to the specified location in the internal memory.

Syntax `:STORe:SETup<x>:COMMent {<String>}`
`:STORe:SETup<x>:COMMent?`
<x> = 1 to 3
<String> = Up to 8 characters

Example
```
:STORE:SETUP1:COMMENT "WAVE1"
:STORE:SETUP1:COMMENT?
-> :STORE:SETUP1:COMMENT "WAVE1"
```

:STORe:SETup<x>:DATE?

Function Queries the date and time of the setup data that is stored to the specified location in the internal memory.

Syntax `:STORe:SETup<x>:DATE?`
<x> = 1 to 3

Example
```
:STORE:SETUP1:DATE?
-> :STORE:SETUP1:DATE "2018/11/11 10
:56:22"
```

:STORe:SETup<x>:EXECute

Function Saves setup data to the specified location in the internal memory.

Syntax `:STORe:SETup<x>:EXECute`
<x> = 1 to 3

Example `:STORE:SETUP1:EXECUTE`

:STORe:SETup<x>:LOCK

Function Sets or queries the protection on/off state for the setup data stored to the specified location in the internal memory.

Syntax `:STORe:SETup<x>:LOCK {<Boolean>}`
`:STORe:SETup<x>:LOCK?`
<x> = 1 to 3

Example
```
:STORE:SETUP1:LOCK ON
:STORE:SETUP1:LOCK?
-> :STORE:SETUP1:LOCK 1
```

5.31 SYSTem Group

:SYSTem?

Function Queries all system settings.

Syntax :SYSTem?

:SYSTem:BEEP

Function Generates a beep sound.

Syntax :SYSTem:BEEP

Example :SYSTEM:BEEP

:SYSTem:CLICk

Function Sets or queries whether click sounds are produced.

Syntax :SYSTem:CLICk {<Boolean>}
:SYSTem:CLICk?

Example :SYSTEM:CLICK ON
:SYSTEM:CLICK?
-> :SYSTEM:CLICK 1

:SYSTem:CLOCk?

Function Queries all date/time settings.

Syntax :SYSTem:CLOCk?

:SYSTem:CLOCk:DATE

Function Sets or queries the date.

Syntax :SYSTem:CLOCk:DATE {<String>}
:SYSTem:CLOCk:DATE?
<String> = YYYY/MM/DD. See the Features Guide for this information.

Example :SYSTEM:CLOCK:DATE "2018/10/17"
:SYSTEM:CLOCK:DATE?
-> :SYSTEM:CLOCK:DATE "2018/10/17"

:SYSTem:CLOCk:FORMat

Function Sets or queries the date format.

Syntax :SYSTem:CLOCk:FORMat {<NRf>}
:SYSTem:CLOCk:FORMat?
<NRf> = 1 to 4

Example :SYSTEM:CLOCK:FORMAT 1
:SYSTEM:CLOCK:FORMAT?
-> :SYSTEM:CLOCK:FORMAT 1

:SYSTem:CLOCk:MODE

Function Sets or queries whether the date and time are displayed.

Syntax :SYSTem:CLOCk:MODE {<Boolean>}
:SYSTem:CLOCk:MODE?

Example :SYSTEM:CLOCK:MODE ON
:SYSTEM:CLOCK:MODE?
-> :SYSTEM:CLOCK:MODE 1

:SYSTem:CLOCk:SNTP?

Function Queries the SNTP clock setting.

Syntax :SYSTem:CLOCk:SNTP?

:SYSTem:CLOCk:SNTP:EXECute

Function Sets the clock using SNTP.

Syntax :SYSTem:CLOCk:SNTP:EXECute

Example :SYSTEM:CLOCK:SNTP:EXECUTE

:SYSTem:CLOCk:SNTP:GMTTime

Function Sets or queries the time difference from Greenwich Mean Time.

Syntax :SYSTem:CLOCk:SNTP:GMTTime {<String>}
:SYSTem:CLOCk:SNTP:GMTTime?
<String> = HH:MM. See the Features Guide for this information.

Example :SYSTEM:CLOCK:SNTP:GMTTIME "09:00"
:SYSTEM:CLOCK:SNTP:GMTTIME?
-> :SYSTEM:CLOCK:SNTP:GMTTIME "09:00"

:SYSTem:CLOCk:TIME

Function Sets or queries the time.

Syntax :SYSTem:CLOCk:TIME {<String>}
:SYSTem:CLOCk:TIME?
<String> = HH:MM:SS. See the Features Guide for this information.

Example :SYSTEM:CLOCK:TIME "14:30:00"
:SYSTEM:CLOCK:TIME?
-> :SYSTEM:CLOCK:TIME "14:30:00"

:SYSTem:DCANcel (Delay Cancel)

Function Sets or queries whether the specified delay is applied to time measurement values.

Syntax :SYSTem:DCANcel {<Boolean>}
:SYSTem:DCANcel?

Example :SYSTEM:DCANCEL ON
:SYSTEM:DCANCEL?
-> :SYSTEM:DCANCEL 1

:SYSTem:LANGuage
Function Sets or queries the message language.
Syntax :SYSTem:LANGuage {CHINese|ENGLish|FRENch|GERMan|ITALian|JAPANese|KORean|RUSSian|SPANish}
:SYSTem:LANGuage?
Example :SYSTEM:LANGUAGE JAPANESE
:SYSTEM:LANGUAGE?
-> :SYSTEM:LANGUAGE JAPANESE

:SYSTem:LCD?
Function Queries all LCD settings.
Syntax :SYSTem:LCD?

:SYSTem:LCD:AUTO?
Function Queries all settings related to the LCD backlight auto-off feature.
Syntax :SYSTem:LCD:AUTO?

:SYSTem:LCD:AUTO:MODE
Function Sets or queries on/off state of the LCD backlight auto-off feature.
Syntax :SYSTem:LCD:AUTO:MODE {<Boolean>}
:SYSTem:LCD:AUTO:MODE?
Example :SYSTEM:LCD:AUTO:MODE ON
:SYSTEM:LCD:AUTO:MODE?
-> :SYSTEM:LCD:AUTO:MODE 1

:SYSTem:LCD:AUTO:TIME
Function Sets or queries the LCD backlight auto-off time.
Syntax :SYSTem:LCD:AUTO:TIME {<NRf>}
:SYSTem:LCD:AUTO:TIME?
<NRf> = 1 to 60 (minutes)
Example :SYSTEM:LCD:AUTO:TIME 1
:SYSTEM:LCD:AUTO:TIME?
-> :SYSTEM:LCD:AUTO:TIME 1

:SYSTem:LCD:BRIGhtness
Function Sets or queries the LCD brightness.
Syntax :SYSTem:LCD:BRIGhtness {<NRf>}
:SYSTem:LCD:BRIGhtness?
<NRf> = 1 to 8
Example :SYSTEM:LCD:BRIGHTNESS 2
:SYSTEM:LCD:BRIGHTNESS?
-> :SYSTEM:LCD:BRIGHTNESS 2

:SYSTem:LCD:MODE
Function Sets or queries the LCD backlight on/off state.
Syntax :SYSTem:LCD:MODE {<Boolean>}
:SYSTem:LCD:MODE?
Example :SYSTEM:LCD:MODE ON
:SYSTEM:LCD:MODE?
-> :SYSTEM:LCD:MODE 1

:SYSTem:MLANguage
Function Sets or queries the menu language.
Syntax :SYSTem:MLANguage {CHINese|ENGLish|FRENch|GERMan|ITALian|JAPANese|KORean|RUSSian|SPANish}
:SYSTem:MLANguage?
Example :SYSTEM:MLANGUAGE JAPANESE
:SYSTEM:MLANGUAGE?
-> :SYSTEM:MLANGUAGE JAPANESE

:SYSTem:MPARameter?
Function Queries all settings related to automated measurement values of waveform parameters and cursor measurement values.
Syntax :SYSTem:MPARameter?

:SYSTem:MPARameter:FSIZe
Function Sets or queries the font size that is used to display the automatically measured waveform parameters and cursor measurement values.
Syntax :SYSTem:MPARameter:FSIZe {LARGe|SMALl}
:SYSTem:MPARameter:FSIZe?
Example :SYSTEM:MPARAMETER:FSIZE SMALl
:SYSTEM:MPARAMETER:FSIZE?
-> :SYSTEM:MPARAMETER:FSIZE SMALl

:SYSTem:MPARameter:MROWs
Function Sets or queries the number of rows of automatically measured waveform parameters and cursor measurement values to show from the menu display area.
Syntax :SYSTem:MPARameter:MROWs {<NRf>}
:SYSTem:MPARameter:MROWs?
<NRf> = 2 to 30
Example :SYSTEM:MPARAMETER:MROWS 4
:SYSTEM:MPARAMETER:MROWS?
-> :SYSTEM:MPARAMETER:MROWS 4

:SYSTem:OCANcel (Offset Cancel)
Function Sets or queries whether the specified offset voltage is applied to measurement results and computation results.
Syntax :SYSTem:OCANcel {<Boolean>}
:SYSTem:OCANcel?
Example :SYSTEM:OCANCEL ON
:SYSTEM:OCANCEL?
-> :SYSTEM:OCANCEL 1

:SYSTem:TOUT?
Function Queries all trigger out settings.

Syntax :SYSTem:TOUT?

:SYSTem:TOUT:POLarity

Function Sets or queries the trigger out polarity.

Syntax :SYSTem:TOUT:POLarity {NEGative|
POSitive}
:SYSTem:TOUT:POLarity?

Example :SYSTEM:TOUT:POLARITY NEGATIVE
:SYSTEM:TOUT:POLARITY?
-> :SYSTEM:TOUT:POLARITY NEGATIVE

:SYSTem:USBKeyboard

Function Sets or queries the USB keyboard type.

Syntax :SYSTem:USBKeyboard {ENGLish|
JAPANese}
:SYSTem:USBKeyboard?

Example :SYSTEM:USBKEYBOARD JAPANESE
:SYSTEM:USBKEYBOARD?
-> :SYSTEM:USBKEYBOARD JAPANESE

5.32 TIMebase Group

:TIMebase?

Function Queries all time base settings.

Syntax `:TIMebase?`

:TIMebase:SRATe? (Sample RATE)

Function Queries the sample rate.

Syntax `:TIMebase:SRATe?`

Example `:TIMEBASE:SRATE?`

`-> :TIMEBASE:SRATE 12.50E+06`

:TIMebase:TDIV

Function Sets or queries the Time/div value.

Syntax `:TIMebase:TDIV {<Time>}`

`:TIMebase:TDIV?`

<Time> = 1ns to 500s

Example `:TIMEBASE:TDIV 2NS`

`:TIMEBASE:TDIV?`

`-> :TIMEBASE:TDIV 2.000E-06`

5.33 TRIGger Group

The following table indicates how the communication commands for trigger types correspond to the trigger types that appear when you press a panel key.

Trigger Type	Communication Command	Setup Menu	
		Panel Key	Trigger Type
Edge	SIMPle	EDGE key	—
OR of multiple edge triggers (edge OR)	OR	ENHANCED key	Edge OR
Combination of multiple inputs (pattern)	PATTern		Pattern
Pulse width	PULSe		Pulse Width
Rise/fall time	RFTime		Rise/Fall Time
Runt	RUNT		Runt
Timeout period	TIMeout		Timeout
Window	WINDow		Window
Window OR	WOR		Window OR
Edge interval trigger	INTerval		Interval
Serial trigger	CAN*		CAN
	CANFD*		CAN FD
	CXPI*		CXPI
	FLEXray*		FlexRay
	I2C*		I2C
	LIN*		LIN
	SENT*		SENT
	SPATtern		User Define
	SPI*		SPI
	UART*		UART
TV	TV:HDTV		HDTV
	TV:NTSC		NTSC
	TV:PAL		PAL
	TV:SDTV		SDTV(480/60p)
	TV:USERdefine		UserDef TV
Combination	COMBination	B TRIG key	—
Forced	FORCe	SHIFT+B TRIG (FORCE TRIG) key	—

* CAN, CANFD, CXPI, FLEXray, I2C, LIN, SENT, SPI, and UART are serial bus options available only to 4-channel models. These commands cannot be used on 2-channel models or 4-channel models without these options.

:TRIGger?

Function Queries all trigger settings.

Syntax `:TRIGger?`

:TRIGger:ABN?

Function Queries all A->B(N) trigger settings.

Syntax `:TRIGger:ABN?`

:TRIGger:ABN:COUNt

Function Sets or queries the number of times condition B must be met for A->B(N) triggers.

Syntax `:TRIGger:ABN:COUNt {<NRf>}`
`:TRIGger:ABN:COUNt?`
<NRf> = 1 to 1000000000

Example
```
:TRIGGER:ABN:COUNT 1
:TRIGGER:ABN:COUNT?
-> :TRIGGER:ABN:COUNT 1
```

:TRIGger:ACTion?

Function Queries all action-on-trigger settings.

Syntax `:TRIGger:ACTion?`

:TRIGger:ACTion:ACQCount

Function Sets or queries the action count of the action-on-trigger.

Syntax `:TRIGger:ACTion:ACQCount {<NRf>|INFinite}`
`:TRIGger:ACTion:ACQCount?`
<NRf> = 1 to 1000000

Example
```
:TRIGGER:ACTION:ACQCOUNT 1
:TRIGGER:ACTION:ACQCOUNT?
-> :TRIGGER:ACTION:ACQCOUNT 1
```

:TRIGger:ACTion:BUZZer
Function Sets or queries whether a warning sound is generated as an action.
Syntax :TRIGger:ACTion:BUZZer {<Boolean>}
:TRIGger:ACTion:BUZZer?
Example :TRIGGER:ACTION:BUZZER ON
:TRIGGER:ACTION:BUZZER?
-> :TRIGGER:ACTION:BUZZER 1

:TRIGger:ACTion:HCOPy
Function Sets or queries whether screen capture data is output as an action.
Syntax :TRIGger:ACTion:HCOPy {<Boolean>}
:TRIGger:ACTion:HCOPy?
Example :TRIGGER:ACTION:HCOPY ON
:TRIGGER:ACTION:HCOPY?
-> :TRIGGER:ACTION:HCOPY 1

:TRIGger:ACTion:MAIL?
Function Queries all settings related to the mail notification on action.
Syntax :TRIGger:ACTion:MAIL?

:TRIGger:ACTion:MAIL:COUNt
Function Sets or queries the upper limit to the number of mail notifications for the mail notification on action.
Syntax :TRIGger:ACTion:MAIL:COUNt {<NRf>}
:TRIGger:ACTion:MAIL:COUNt?
<NRf> = 1 to 1000
Example :TRIGGER:ACTION:MAIL:COUNT 1
:TRIGGER:ACTION:MAIL:COUNT?
-> :TRIGGER:ACTION:MAIL:COUNT 1

:TRIGger:ACTion:MAIL:MODE
Function Sets or queries whether a mail notification is made as an action.
Syntax :TRIGger:ACTion:MAIL:MODE {<Boolean>}
:TRIGger:ACTion:MAIL:MODE?
Example :TRIGGER:ACTION:MAIL:MODE ON
:TRIGGER:ACTION:MAIL:MODE?
-> :TRIGGER:ACTION:MAIL:MODE 1

:TRIGger:ACTion:SAVE
Function Sets or queries whether waveform data is saved to the storage device as an action.
Syntax :TRIGger:ACTion:SAVE {<Boolean>}
:TRIGger:ACTion:SAVE?
Example :TRIGGER:ACTION:SAVE ON
:TRIGGER:ACTION:SAVE?
-> :TRIGGER:ACTION:SAVE 1

:TRIGger:ACTion:STARt
Function Starts action-on-trigger.
Syntax :TRIGger:ACTion:STARt
:TRIGGER:ACTION:START
Description To abort the trigger action, use ":TRIGger:ACTION:STOP".

:TRIGger:ACTion:STOP
Function Stops action-on-trigger.
Syntax :TRIGger:ACTion:STOP
:TRIGGER:ACTION:STOP
Description This command only stops the trigger action. It does not cancel the setting.
Use ":TRIGger:ACTION:STARt" to resume the trigger action.

:TRIGger:ADB?
Function Queries all A Delay B trigger settings.
Syntax :TRIGger:ADB?

:TRIGger:ADB:DELay
Function Sets or queries the delay time for condition B for A Delay B triggers.
Syntax :TRIGger:ADB:DELay {<Time>}
:TRIGger:ADB:DELay?
<Time> = 10 ns to 10 s (in 2 ns steps)
Example :TRIGGER:ADB:DELAY 10ns
:TRIGGER:ADB:DELAY?
-> :TRIGGER:ADB:DELAY 10ns

:TRIGger:{ATRigger|BTRigger}?
Function Queries all trigger condition settings.
Syntax :TRIGger:{ATRigger|BTRigger}?

:TRIGger{[:ATRigger]|:BTRigger}:CAN?
Function Queries all CAN bus signal trigger settings.
Syntax :TRIGger{[:ATRigger]|:BTRigger}:CAN?

:TRIGger{[:ATRigger]|:BTRigger}:CAN:BRATe

Function Sets or queries the CAN bus signal trigger bit rate (data transfer rate).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
BRATe {<NRf>|USER,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
BRATe?
```
<NRf> = 33300, 83300, 125000, 250000, 500000, 1000000
USER <NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:CAN:BRATE 83300
:TRIGGER:ATRIGGER:CAN:BRATE?
-> :TRIGGER:ATRIGGER:CAN:BRATE 83300
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame?

Function Queries all CAN bus signal trigger error settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
EFRame?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame:CRC

Function Sets or queries the CAN bus signal trigger CRC error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
EFRame:CRC {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
EFRame:CRC?
```

Example
```
:TRIGGER:ATRIGGER:CAN:EFRAME:CRC ON
:TRIGGER:ATRIGGER:CAN:EFRAME:CRC?
-> :TRIGGER:ATRIGGER:CAN:EFRAME:C
RC 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame[:MODE]

Function Sets or queries the CAN bus signal trigger error frame setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
EFRame[:MODE] {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
EFRame:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:EFRAME:MODE ON
:TRIGGER:ATRIGGER:CAN:EFRAME:MODE?
-> :TRIGGER:ATRIGGER:CAN:EFRAME:MOD
E 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame:STUFF

Function Sets or queries the CAN bus signal trigger stuff error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
EFRame:STUFF {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
EFRame:STUFF?
```

Example
```
:TRIGGER:ATRIGGER:CAN:EFRAME:STU
FF ON
:TRIGGER:ATRIGGER:CAN:EFRAME:STUFF?
-> :TRIGGER:ATRIGGER:CAN:EFRAME:STU
FF 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDData?

Function Queries all ID and data condition settings for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:
IDData?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK?

Function Queries the ACK setting, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:ACK?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK:MODE

Function Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:ACK:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:ACK:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:ACK:MO
DE ON
:TRIGGER:ATRIGGER:CAN:IDDATA:ACK:MO
DE?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:ACK:
MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK:TYPE

Function Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK:TYPE {ACK|ACKBoth|NONack}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:ACK:TYPE ACK
:TRIGGER:ATRIGGER:CAN:IDDATA:ACK:TYPE?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:ACK:TYPE ACK
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA?

Function Queries the data, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:CONDition

Function Sets or queries the data comparison condition, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:CONDITION EQUAL
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DECimal<x>

Function Sets a reference value, which is one of the ID and data conditions, for CAN bus signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DECimal<x>?
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DECIMAL1 1
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DECIMAL1?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DECIMAL1 1.000E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DLC

Function Sets or queries the number of significant bytes (DLC), which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DLC {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DLC?
```
<NRf> = 0 to 8

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DLC 0
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DLC?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DLC 0
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DPATtern:HEXa

Function Sets a reference value, which is one of the ID and data conditions, for CAN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DPATtern:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DLC setting.

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DPATtern:PATTern

Function Sets or queries the data, which is one of the ID and data conditions, for CAN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DPATtern:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DPATtern:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DPATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DLC setting.

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:ENDian

Function Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:ENDian {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:ENDian?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:ENDIAN BIG
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:ENDIAN?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MODE

Function Sets or queries the data condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MODE ON
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MSBLsb

Function Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MSBLsb?
```
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MSBLSB 1,0
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MSBLSB?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MSBLSB 1,0
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:SIGN

Function Sets or queries the data sign, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:SIGN {SIGN|UNSign}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:SIGN?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:SIGN SIGN
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:SIGN?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:SIGN SIGN
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier?

Function Queries the identifier, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID?

Function Queries the ID, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:HEXa

Function Sets an ID, which is one of the ID and data conditions, for CAN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:HEXa{<String>}
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFIER:ID:HEXA "1AB"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:MODE

Function Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:CAN[:IDData]:IDENtifier:ID:MO
DE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:IDENtifier:ID<x>:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:ID:MODE ON
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:ID:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDEN
TIFIER:ID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:PATTern

Function Sets or queries the ID condition, which is one of the ID and data conditions, for CAN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CAN[:IDData]:IDENtifier:ID:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:IDENtifier:ID:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:ID:PATTERN "11100001111"
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:ID:PATTERN?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDEN
TIFIER:ID:PATTERN "11100001111"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:CAN[:IDData]:IDENtifier:MFORm
at {EXTended|STANdard}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:IDENtifier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:MFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDEN
TIFIER:MFORMAT EXTENDED
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:CAN[:IDData]:IDENtifier:PFORm
at {MESSage|PATTern}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:PFORMAT PATTERN
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIF
IER:PFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDEN
TIFIER:PFORMAT PATTERN
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal?

Function Queries the message signal, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:MSIGnal?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:MESSage:ITEM

Function Sets the message item, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CAN[:IDData]:MSIGnal:MESSage:IT
EM {<String>}
```
<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL
:MESSAGE:ITEM "TEST"
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SELect

Function Sets or queries the CAN bus signal trigger message signal setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigge
r}:CAN[:IDData]:MSIGnal:SELe
ct {MESSage|SIGNal}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[
:IDData]:MSIGnal:SELect?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL
:SELECT MESSAGE
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL
:SELECT?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:MSIG
NAL:SELECT MESSAGE
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal?

Function Queries the signal, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:CONDition

Function Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:SIGNAL:CONDITION EQUAL
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:SIGNAL:CONDITION?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:SIGNAL:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:DECimal<x>

Function Sets a reference signal value, which is one of the ID and data conditions, for CAN bus signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:DECimal<x>{<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:DECimal<x>?
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:SIGNAL:DECIMAL1 1
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:SIGNAL:DECIMAL1?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:SIGNAL:DECIMAL1 1.000E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:ITEM

Function Sets the signal item, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:ITEM {<String>,<String>}
```
<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:SIGNAL:ITEM "Sig_Test","Mess_Test"
```

Description Set the signal item first and then the message item.

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:RTR

Function Sets or queries the RTR value, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:RTR {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:RTR?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:RTR ON
:TRIGGER:ATRIGGER:CAN:IDDATA:RTR?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:RTR 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR?

Function Queries all ID OR condition settings for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK?

Function Queries all ACK settings for the ID OR conditions for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:MODE

Function Sets or queries the ACK mode, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:MODE ON
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDOR:ACK:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:TYPE

Function Sets or queries the ACK condition, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:TYPE {ACK|ACKBoth|NONack}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:TYPE ACK
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:TYPE?
-> :TRIGGER:ATRIGGER:CAN:IDOR:ACK:TYPE ACK
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA?

Function Queries all data settings for the ID OR conditions for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA[:MODE]

Function Sets or queries the data enable/disable condition, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA[:MODE] {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA[:MODE]?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:DATA:MODE ON
:TRIGGER:ATRIGGER:CAN:IDOR:DATA:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDOR:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier?

Function Queries all identifier settings for the ID OR conditions for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>?

Function Queries all ID settings for the ID OR conditions for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:HEXa

Function Sets the ID value, which is one of the ID OR conditions, for CAN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:ID1:HEXA "1AB"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:ID1:MODE ON
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:ID1:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:ID1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:PATTern

Function Sets or queries the ID condition, which is one of the ID OR conditions, for CAN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:PATTern?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:ID1:PATTERN "11100001111"
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:ID1:PATTERN?
-> :TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:ID1:PATTERN "11100001111"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MODE

Function Sets or queries the identifier enable/disable condition, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MODE {<Boolean>}
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:MODE ON
-> :TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MFORmat {EXTended|STANdard}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:MFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:MFORMAT EXTENDED
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:PFORmat {MESSage|PATTern}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:PFORMAT PATTERN
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:PFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:PFORMAT PATTERN
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:MSIGnal<x>:MESSage:ITEM

Function Sets the message item, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:MSIGnal<x>:MESSage:ITEM {<String>}
```
<x> = 1 to 4
<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:MSIGNAL1:MESSAGE:ITEM "TEST"
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:RTR

Function Sets or queries the RTR, which is one of the ID OR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:RTR {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:RTR?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:RTR ON
:TRIGGER:ATRIGGER:CAN:IDOR:RTR?
-> :TRIGGER:ATRIGGER:CAN:IDOR:RTR 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:MODE

Function Sets or queries the CAN bus signal trigger mode.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:MODE {EFRame|IDData|IDOR|SOF}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:MODE EFRAME
:TRIGGER:ATRIGGER:CAN:MODE?
-> :TRIGGER:ATRIGGER:CAN:MODE EFRAME
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:RECessive

Function Sets or queries the CAN bus signal trigger recessive level.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:RECessive {HIGH|LOW}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:RECessive?
```

Example
```
:TRIGGER:ATRIGGER:CAN:RECESSIVE HIGH
:TRIGGER:ATRIGGER:CAN:RECESSIVE?
-> :TRIGGER:ATRIGGER:CAN:RECESSIVE HIGH
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:SOURce

Function Sets or queries the CAN bus signal trigger signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CAN:SOURCE 1
:TRIGGER:ATRIGGER:CAN:SOURCE?
-> :TRIGGER:ATRIGGER:CAN:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:SPOint

Function Sets or queries the CAN bus signal trigger sample point.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:SPOint {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:SPOint?
```
<NRf> = 18.8 to 90.6

Example
```
:TRIGGER:ATRIGGER:CAN:SPOINT 18.8
:TRIGGER:ATRIGGER:CAN:SPOINT?
-> :TRIGGER:ATRIGGER:CAN:SPOINT 18.8
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD?

Function Queries all CAN FD bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN
FD?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:BRATe

Function Sets or queries the CAN FD bus signal trigger bit rate (data transfer rate).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:BRATe {<NRf>|USER,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:BRATe?
```
<NRf> = 250000, 500000, 1000000
USER <NRf> = 20000 to 1000000 (resolution: 100)

Example
```
:TRIGGER:ATRIGGER:CANFD:BRATE 500000
:TRIGGER:ATRIGGER:CANFD:BRATE?
-> :TRIGGER:ATRIGGER:CANFD:BRATE 500
000
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:DBRate

Function Sets or queries the CAN FD bus signal trigger data phase bit rate (data transfer rate).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:DBRate {<NRf>|USER,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:DBRate?
```
<NRf> = 500000, 1000000, 2000000, 4000000, 5000000, 8000000
USER <NRf> = 250000 to 10000000 (resolution: 100)

Example
```
:TRIGGER:ATRIGGER:CANFD:DBRATE 10000
00
:TRIGGER:ATRIGGER:CANFD:DBRATE?
-> :TRIGGER:ATRIGGER:CANFD:DBRATE 10
00000
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:DSPoint

Function Sets or queries the CAN FD bus signal trigger data phase sample point.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:DSPoint {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:DSPoint?
```
<NRf> = 18.8 to 90.6 (resolution: 0.1)

Example
```
:TRIGGER:ATRIGGER:CANFD:DSPOINT 18.8
:TRIGGER:ATRIGGER:CANFD:DSPOINT?
-> :TRIGGER:ATRIGGER:CANFD:DSPOINT 1
8.8
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame?

Function Queries all CAN FD bus signal trigger error settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:EFRame?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:CRC

Function Sets or queries the CAN FD bus signal trigger CRC error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:EFRame:CRC {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:EFRame:CRC?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:EFRAME:C
RC ON
:TRIGGER:ATRIGGER:CANFD:EFRAME:CRC?
-> :TRIGGER:ATRIGGER:CANFD:EFRAME:C
RC 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:CRCEFactor?

Function Queries all CRC error factor settings for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:EFRame:CRCEFactor?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:CRCEFactor:CRCSequence

Function Sets or queries the CRC sequence (a CRC error factor) for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:C
ANFD:EFRame:CRCEFactor:CRCSequen
ce {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:EFRame:CRCEFactor:CRCSequence?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:EFRAME:CRCEF
ACTOR:CRCSEQUENCE ON
:TRIGGER:ATRIGGER:CANFD:EFRAME:CRCEF
ACTOR:CRCSEQUENCE?
-> :TRIGGER:ATRIGGER:CANFD:EFRAME:CR
CEFACTOR:CRCSEQUENCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:CRCEFactor:SCOunt

Function Sets or queries the Stuff Count (a CRC error factor) for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:CRCEFactor:SCOunt {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:CRCEFactor:SCOunt?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:EFRAME:CRCEFACTOR:SCOUNT ON
:TRIGGER:ATRIGGER:CANFD:EFRAME:CRCEFACTOR:SCOUNT?
-> :TRIGGER:ATRIGGER:CANFD:EFRAME:CRCEFACTOR:SCOUNT 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:FSTuff

Function Sets or queries the CAN FD bus signal trigger fixed stuff error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:FSTuff {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:FSTuff?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:EFRAME:FSTUFF ON
:TRIGGER:ATRIGGER:CANFD:EFRAME:FSTUFF?
-> :TRIGGER:ATRIGGER:CANFD:EFRAME:FSTUFF 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame[:MODE]

Function Sets or queries the CAN FD bus signal trigger error frame setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame[:MODE] {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:Eframe:mode?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:EFRAME:MODE ON
:TRIGGER:ATRIGGER:CANFD:EFRAME:MODE?
-> :TRIGGER:ATRIGGER:CANFD:EFRAME:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:STUFF

Function Sets or queries the CAN FD bus signal trigger stuff error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:STUFF {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:EFRame:STUFF?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:EFRAME:STUFF ON
:TRIGGER:ATRIGGER:CANFD:EFRAME:STUFF?
-> :TRIGGER:ATRIGGER:CANFD:EFRAME:STUFF 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:FDF:CONDition

Function Sets or queries the CAN FD bus signal trigger FDF.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:FDF:CONDition {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:FDF:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:EFRAME:FDF:CONDITION ON
:TRIGGER:ATRIGGER:CANFD:EFRAME:FDF:CONDITION?
-> :TRIGGER:ATRIGGER:CANFD:EFRAME:FDF:CONDITION 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:FDSTandard

Function Sets or queries whether the CAN FD bus signal for triggering is an ISO standard signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:FDSTandard {ISO|NISO}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:FDSTandard?
```

Example
```
:TRIGger:ATRIGGER:CANFD:FDSTANDARD ISO
:TRIGger:ATRIGGER:CANFD:FDSTANDARD?
-> :TRIGger:ATRIGGER:CANFD:FDSTANDARD ISO
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDData?

Function Queries all ID and data condition settings for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDData?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK?

Function Queries all ACK settings for the ID and data conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK:MODE

Function Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:ACK:MODE ON
:TRIGGER:ATRIGGER:CANFD:IDDATA:ACK:MODE?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:ACK:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK:TYPE

Function Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK:TYPE {ACK|ACKBoth|NONack}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:ACK:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:ACK:TYPE ACK
:TRIGGER:ATRIGGER:CANFD:IDDATA:ACK:TYPE?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:ACK:TYPE ACK
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA?

Function Queries all data settings for the ID and data conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:BCOunt

Function Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:BCOUNT {<NRf>}
```
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:BCOUNT 4
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:BCOUNT?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:BCOUNT 4
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:CONDition

Function Sets or queries the comparison condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:CONDITION EQUAL
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DBYTe

Function Sets or queries the number of data bytes, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DBYTe?
```
<NRf> = 0 to 8

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DBYTE 2
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DBYTE?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DBYTE 2
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DECimal<x>

Function Sets a reference values (a, b), which is one of the ID and data conditions, for CAN FD bus signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DECimal<x>?
```
<x> = 1 or 2
<NRf> = –9E+18 to 9E+18

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DECIMAL1 1
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DECIMAL1?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DECIMAL1 1.000E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DPATtern:HEXa

Function Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DPATtern:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DPATtern:PATTern

Function Sets or queries the data value, which is one of the ID and data conditions, for CAN FD bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DPATtern:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DPATtern:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DPATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:ENDian

Function Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:ENDian {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:ENDian?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:ENDIAN BIG
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:ENDIAN?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:MODE

Function Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:MODE ON
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:MODE?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:MSBLsb

Function Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:MSBLsb?
```
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:MSBLSB 1,0
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:MSBLSB?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:MSBLSB 1,0
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:SIGN

Function Sets or queries whether signed or unsigned data format will be used, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:SIGN {SIGN|UNSign}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:DATA:SIGN?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:SIGN SIGN
:TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:SIGN?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:DATA:SIGN SIGN
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier?

Function Queries all identifier settings for the ID and data conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID?

Function Queries all identifier settings for the ID and data conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID:HEXa

Function Sets the ID value, which is one of the ID and data conditions, for CAN FD bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:ID:HEXA "1AB"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID<x>:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:ID:MODE ON
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:ID:MODE?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:ID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID:PATTern

Function Sets or queries the ID condition, which is one of the ID and data conditions, for CAN FD bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:ID:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:ID:PATTERN "11100001111"
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:ID:PATTERN?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:ID:PATTERN "11100001111"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:MFORmat {EXTended|STANdard}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:MFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:IDENTIFIER:MFORMAT EXTENDED
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CANFD[:IDData]:IDENtifier:PFORm
at {MESSage|PATTern}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D[:IDData]:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENT
IFIER:PFORMAT PATTERN
:TRIGGER:ATRIGGER:CANFD:IDDATA:IDENT
IFIER:PFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:ID
ENTIFIER:PFORMAT PATTERN
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:MSIGnal?

Function Queries all message and signal settings for the ID and data conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D[:IDData]:MSIGnal?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:MSIGnal:MESSage:ITEM

Function Sets the message item, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:C
ANFD[:IDData]:MSIGnal:MESSage:IT
EM {<String>}
```
<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:MSIGN
AL:MESSAGE:ITEM "TEST"
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:MSIGnal:SELect

Function Sets or queries the message and signal condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger
}:CANFD[:IDData]:MSIGnal:SELe
ct {MESSage|SIGNal}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D[:IDData]:MSIGnal:SELect?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:MSIGN
AL:SELECT MESSAGE
:TRIGGER:ATRIGGER:CANFD:IDDATA:MSIGN
AL:SELECT?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:MS
IGNAL:SELECT MESSAGE
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:MSIGnal:SIGNal?

Function Queries all signal settings for the ID and data conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D[:IDData]:MSIGnal:SIGNal?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:MSIGnal:SIGNal:CONDition

Function Sets or queries the signal condition, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN
FD[:IDData]:MSIGnal:SIGNal:CONditi
on {EQUal|GREater|INSide|LESS|NOTEqu
al|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D[:IDData]:MSIGnal:SIGNal:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:MSIGN
AL:SIGNAL:CONDITION EQUAL
:TRIGGER:ATRIGGER:CANFD:IDDATA:MSIGN
AL:SIGNAL:CONDITION?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:MS
IGNAL:SIGNAL:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:MSIGnal:SIGNal:DECimal<x>

Function Sets a signal's reference value, which is one of the ID and data conditions, for CAN FD bus signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN
FD[:IDData]:MSIGnal:SIGNal:DECimal
<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN
FD[:IDData]:MSIGnal:SIGNal:DECimal
<x>?
```
<x> = 1 or 2
<NRf> = –9E+18 to 9E+18

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:MSIGN
AL:SIGNAL:DECIMAL1 1
:TRIGGER:ATRIGGER:CANFD:IDDATA:MSIGN
AL:SIGNAL:DECIMAL1?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:MS
IGNAL:SIGNAL:DECIMAL1 1.000E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:MSIGnal:SIGNal:ITEM

Function Sets the signal item, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CANFD[:IDData]:MSIGnal:SIGNal:IT
EM {<String>,<String>}
```
<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CANFD:
IDDATA:MSIGNAL:SIGNAL:IT
EM "Sig_Test","Mess_Test"
```

Description Set the signal item first and then the message item.

:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:RTR

Function Sets or queries the RTR, which is one of the ID and data conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:RTR {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD[:IDData]:RTR?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDDATA:RTR ON
:TRIGGER:ATRIGGER:CANFD:IDDATA:RTR?
-> :TRIGGER:ATRIGGER:CANFD:IDDATA:RTR 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR?

Function Queries all ID OR condition settings for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK?

Function Queries all ACK settings for the ID OR conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK:MODE

Function Sets or queries the ACK mode, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:ACK:MODE ON
:TRIGGER:ATRIGGER:CANFD:IDOR:ACK:MODE?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:ACK:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK:TYPE

Function Sets or queries the ACK condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK:TYPE {ACK|ACKBoth|NONack}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:ACK:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:ACK:TYPE ACK
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:TYPE?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:ACK:TYPE ACK
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:DATA?

Function Queries all data settings for the ID OR conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:DATA[:MODE]

Function Sets or queries the data enable/disable condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:DATA[:MODE] {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:DATA[:MODE]?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:DATA:MODE ON
:TRIGGER:ATRIGGER:CANFD:IDOR:DATA:MODE?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier?

Function Queries all identifier settings for the ID OR conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:ID<x>?

Function Queries all ID settings for the ID OR conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:ID<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:ID<x>:HEXa

Function Sets the ID value, which is one of the ID OR conditions, for CAN FD bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:CANFD:IDOR:IDENtifier:ID<x>:HE
Xa {<String>}
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:ID1:HEXA "1AB"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:ID<x>:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:CANFD:IDOR:IDENtifier:ID<x>:MO
DE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:IDOR:IDENtifier:ID<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:ID1:MODE ON
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:ID1:MODE?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDEN
TIFIER:ID1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:ID<x>:PATTern

Function Sets or queries the ID condition, which is one of the ID OR conditions, for CAN FD bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:C
ANFD:IDOR:IDENtifier:ID<x>:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:IDOR:IDENtifier:ID<x>:PATTern?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:ID1:PATTERN "11100001111"
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:ID1:PATTERN?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDEN
TIFIER:ID1:PATTERN "11100001111"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:MFORmat setting.

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger
}:CANFD:IDOR:IDENtifier:MFORm
at {EXTended|STANdard}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:IDOR:IDENtifier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:MFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDEN
TIFIER:MFORMAT EXTENDED
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:IDOR:IDENtifier:MODE {<Boolean>}
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:MODE ON
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDEN
TIFIER:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger
}:CANFD:IDOR:IDENtifier:PFORm
at {MESSage|PATTern}
:TRIGger{[:ATRigger]|:BTRigger}:CANF
D:IDOR:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:PFORMAT PATTERN
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIF
IER:PFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDEN
TIFIER:PFORMAT PATTERN
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:MSIGnal<x>:MESSage:ITEM

Function Sets the message item, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CANFD:IDOR:MSIGnal<x>:MESSage:IT
EM {<String>}
```
<x> = 1 to 4
<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:MSIGNAL1:
MESSAGE:ITEM "TEST"
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:RTR

Function Sets or queries the RTR, which is one of the ID OR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:RTR {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:RTR?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:RTR ON
:TRIGGER:ATRIGGER:CANFD:IDOR:RTR?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:RTR 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:MODE

Function Sets or queries the CAN FD bus signal trigger mode.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:MODE {EFRame|ESI|FDF|IDData|IDOR|SOF}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:MODE EFRAME
:TRIGGER:ATRIGGER:CANFD:MODE?
-> :TRIGGER:ATRIGGER:CANFD:MODE EFRAME
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:RECessive

Function Sets or queries the CAN FD bus signal trigger recessive level.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:RECessive {HIGH|LOW}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:RECessive?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:RECESSIVE HIGH
:TRIGGER:ATRIGGER:CANFD:RECESSIVE?
-> :TRIGGER:ATRIGGER:CANFD:RECESSIVE HIGH
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:SOURce

Function Sets or queries the CAN FD bus signal trigger signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:SOURCE 1
:TRIGGER:ATRIGGER:CANFD:SOURCE?
-> :TRIGGER:ATRIGGER:CANFD:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CANFD:SPOint

Function Sets or queries the CAN FD bus signal trigger sample point.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:SPOint {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:SPOint?
```
<NRf> = 18.8 to 90.6 (resolution: 0.1)

Example
```
:TRIGGER:ATRIGGER:CANFD:SPOINT 18.8
:TRIGGER:ATRIGGER:CANFD:SPOINT?
-> :TRIGGER:ATRIGGER:CANFD:SPOINT 18.8
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI?

Function Queries all CXPI bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:BRATe

Function Sets or queries the CXPI bus signal trigger bit rate (data transfer rate).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI:BRATe {<NRf>|USER,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI:BRATe?
```
<NRf> = 4800, 9600, 19200
USER <NRf> = 4000 to 50000

Example
```
:TRIGGER:ATRIGGER:CXPI:BRATE 19200
:TRIGGER:ATRIGGER:CXPI:BRATE?
-> :TRIGGER:ATRIGGER:CXPI:BRATE 19200
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:CTOLerance

Function Sets or queries the CXPI bus signal trigger clock tolerance.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI:CTOLerance {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI:CTOLerance?
```
<NRf> = ±0.5% to ±5.0% (resolution: 0.1%)

Example
```
:TRIGGER:ATRIGGER:CXPI:CTOLERANCE 0.04
:TRIGGER:ATRIGGER:CXPI:CTOLERANCE?
-> :TRIGGER:ATRIGGER:CXPI:CTOLERANCE 0.04
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:ERRor?

Function Queries all CXPI bus signal trigger error settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:ERRor:CLOCk

Function Sets or queries the CXPI bus signal trigger clock error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:CLOCk {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:CLOCk?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:ERROR:CLO
CK ON
:TRIGGER:ATRIGGER:CXPI:ERROR:CLOCK?
-> :TRIGGER:ATRIGGER:CXPI:ERROR:CLO
CK 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:ERRor:CRC

Function Sets or queries the CXPI bus signal trigger CRC error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:CRC {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:CRC?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:ERROR:CRC ON
:TRIGGER:ATRIGGER:CXPI:ERROR:CRC?
-> :TRIGGER:ATRIGGER:CXPI:ERROR:C
RC 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:ERRor:DLENgth

Function Sets or queries the CXPI bus signal trigger data length error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:DLENgth {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:DLENgth?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:ERROR:DLENG
TH ON
:TRIGGER:ATRIGGER:CXPI:ERROR:DLENG
TH?
-> :TRIGGER:ATRIGGER:CXPI:ERROR:DLEN
GTH 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:ERRor:FRAMing

Function Sets or queries the CXPI bus signal trigger framing error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:FRAMing {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:FRAMing?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:ERROR:FRAMI
NG ON
:TRIGGER:ATRIGGER:CXPI:ERROR:FRAMI
NG?
-> :TRIGGER:ATRIGGER:CXPI:ERROR:FRAM
ING 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:ERRor:IBS

Function Sets or queries the CXPI bus signal trigger IBS error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:IBS {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:IBS?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:ERROR:IBS ON
:TRIGGER:ATRIGGER:CXPI:ERROR:IBS?
-> :TRIGGER:ATRIGGER:CXPI:ERROR:I
BS 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:ERRor:PARity

Function Sets or queries the CXPI bus signal trigger parity error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:PARity {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:ERRor:PARity?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:ERROR:PARI
TY ON
:TRIGGER:ATRIGGER:CXPI:ERROR:PARITY?
-> :TRIGGER:ATRIGGER:CXPI:ERROR:PARI
TY 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData?

Function Queries all ID and data condition settings for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA?

Function Queries the data, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:BCOunt

Function Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:BCOunt {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:BCOunt?
```
<NRf> = 0 to 254

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:B
COUNT 1
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:B
COUNT?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:BCOUNT 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:CONDition

Function Sets or queries the data comparison condition, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:CONDition {EQUal|FALSE|
GREater|INSide|LESS|NOTEqual|OUTSide
|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:CONDition?
```
Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:C
ONDITION EQUAL
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:C
ONDITION?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:DBYTe

Function Sets or queries the number of data bytes, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:DBYTe?
```
<NRf> = 0 to 8

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:D
BYTE 1
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:D
BYTE?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:DBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:DECimal<x>

Function Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:DECimal<x>?
```
<x> = 1 or 2

<NRf> = –9E+18 to 9E+18

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:D
ECIMAL1 1
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:D
ECIMAL1?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:DECIMAL1 1.0000000E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:DPATtern:HEXa

Function Sets an ID, which is one of the ID and data conditions, for CXPI bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:DPATtern:HEXa {<String>}
```
Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:D
PATTERN:HEXA "12"
```
Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:DPATtern:PATTern

Function Sets or queries the ID condition, which is one of the ID and data conditions, for CXPI bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CXPI:IDData:DATA:DPATtern:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:DPATtern:PATTern?
```
Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:D
PATTERN:PATTERN "00110101"
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:D
PATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:DPATTERN:PATTERN "00110101"
```
Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:ENDian

Function Sets or queries the byte order of the reference values, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:ENDian {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:ENDian?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:E
NDIAN BIG
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:E
NDIAN?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:MODE

Function Sets or queries the ID and data condition (enabled/disabled) for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:M
ODE ON
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:M
ODE?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:MSBLsb

Function Sets or queries the MSB and LSB bits for the data, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:MSBLsb {<NRf>,<NRf>}
```
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:M
SBLSB 7,0
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:M
SBLSB?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:MSBLSB 7,0
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:DATA:SIGN

Function Sets or queries the data sign, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:SIGN {SIGN|UNSign}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:DATA:SIGN?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:S
IGN SIGN
:TRIGGER:ATRIGGER:CXPI:IDDATA:DATA:S
IGN?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:DAT
A:SIGN SIGN
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:FINFormation

Function Queries all frame information settings for the ID and data conditions for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:FINFormation?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:FINFormation:CT

Function Sets or queries the frame information counter value, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigge
r}:CXPI:IDData:FINFormation:
CT {<NRf>|DONTcare}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:FINFormation:CT?
```
<NRf> = 0 to 3

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:CT 3
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:CT?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:FIN
FOMATION:CT 3
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:FINFormation:MODE

Function Sets or queries the frame information (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger
}:CXPI:IDData:FINFormation:MO
DE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:FINFormation:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:MODE ON
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:MODE?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:FIN
FOMATION:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:FINFormation:SLEEP

Function Sets or queries the frame information sleep bit, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:FINFormation:SLEEP {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:FINFormation:SLEEP?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:SLEEP 1
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:SLEEP?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:FIN
FOMATION:SLEEP 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:FINFormation:WAKeup

Function Sets or queries the frame information wakeup bit, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:FINFormation:WAKeup {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:FINFormation:WAKeup
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:WAKEUP 1
:TRIGGER:ATRIGGER:CXPI:IDDATA:FINFOM
ATION:WAKEUP?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:FIN
FOMATION:WAKEUP 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:ID?

Function Queries the ID, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:ID:HEXa

Function Sets an ID, which is one of the ID and data conditions, for CXPI bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:ID:HE
XA "1E"
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:ID:MODE

Function Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:ID:MO
DE ON
:TRIGGER:ATRIGGER:CXPI:IDDATA:ID:MO
DE?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:ID:
MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:ID:PATTern

Function Sets or queries the ID, which is one of the ID and data conditions, for CXPI bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:ID:PAT
TERN "0010000"
:TRIGGER:ATRIGGER:CXPI:IDDATA:ID:PAT
TERN?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:ID:
PATTERN "0010000"
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDData:ID:PTYPE

Function Sets or queries the PTYPE comparison condition, which is one of the ID and data conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID:PTYPE {DONTcare|NO|YES}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDData:ID:PTYPE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDDATA:ID:PTY
PYE DONTCARE
:TRIGGER:ATRIGGER:CXPI:IDDATA:ID:PTY
PYE?
-> :TRIGGER:ATRIGGER:CXPI:IDDATA:ID:
PTYPYE DONTCARE
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR?

Function Queries all CXPI bus signal trigger IDOR condition settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR:IDENtifier?

Function Queries all identifier settings for the IDOR conditions for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR:IDENtifier:ID<x>?

Function Queries all ID settings for the IDOR conditions for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier:ID<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR:IDENtifier:ID<x>:HEXa

Function Sets the ID value, which is one of the IDOR conditions, for CXPI bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier:ID<x>:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:ID1:HEXA "1E"
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR:IDENtifier:ID<x>:MODE

Function Sets or queries the ID enable/disable condition, which is one of the IDOR conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:CXPI:IDOR:IDENtifier:ID<x>:MO
DE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier:ID<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:ID1:MODE ON
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:ID1:MODE?
-> :TRIGGER:ATRIGGER:CXPI:IDOR:IDENT
IFIER:ID1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR:IDENtifier:ID<x>:PATTern

Function Sets or queries the ID condition, which is one of the IDOR conditions, for CXPI bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CXPI:IDOR:IDENtifier:ID<x>:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier:ID<x>:PATTern?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:ID1:PATTERN "0010000"
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:ID1:PATTERN?
-> :TRIGGER:ATRIGGER:CXPI:IDOR:IDENT
IFIER:ID1:PATTERN "0010000"
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR:IDENtifier:MODE

Function Sets or queries the identifier enable/disable condition, which is one of the IDOR conditions, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:MODE ON
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:MODE?
-> :TRIGGER:ATRIGGER:CXPI:IDOR:IDENT
IFIER:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:IDOR:IDENtifier:PTYPE

Function Sets or queries the comparison condition of PTYPE, which is an IDOR condition, for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigg
er}:CXPI:IDOR:IDENtifier:PTY
PE {DONTcare|NO|YES}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:IDOR:IDENtifier:PTYPE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:PTYPE DONTCARE
:TRIGGER:ATRIGGER:CXPI:IDOR:IDENTIFI
ER:PTYPE?
-> :TRIGGER:ATRIGGER:CXPI:IDOR:IDENT
IFIER:PTYPE DONTCARE
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:MODE

Function Sets or queries the CXPI bus signal trigger type.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:MODE {ERRor|IDData|IDOR|PTYPE|SOF|W
AKeupsleep}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:MODE SOF
:TRIGGER:ATRIGGER:CXPI:MODE?
-> :TRIGGER:ATRIGGER:CXPI:MODE SOF
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:PTYPE?

Function Queries all CXPI bus signal trigger PTYPE settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:PTYPE?
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:PTYPE:CONDition

Function Sets or queries the CXPI bus signal trigger PTYPE condition setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:PTYPE:CONDition {EPTYPe|NOResponse}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:PTYPE:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:PTYPE:CONDITI
ON EPTYPE
:TRIGGER:ATRIGGER:CXPI:PTYPE:CONDITI
ON?
-> :TRIGGER:ATRIGGER:CXPI:PTYPE:COND
ITION EPTYPE
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:SOURce

Function Sets or queries the CXPI bus signal trigger signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CXPI:SOURCE 1
:TRIGGER:ATRIGGER:CXPI:SOURCE?
-> :TRIGGER:ATRIGGER:CXPI:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:TSAMple

Function Sets or queries the logic value (1 or 0) determination threshold for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:TSAMple <NRf>
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:TSAMple?
```
<NRf> = 0.01 Tbit to 0.3 Tbit (resolution: 0.001 Tbit)

Example
```
:TRIGGER:ATRIGGER:CXPI:TSAMPLE 0.04
:TRIGGER:ATRIGGER:CXPI:TSAMPLE?
-> :TRIGGER:ATRIGGER:CXPI:TSAMPLE 0.
04
```

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:WAKeupsleep

Function Queries all wakeup and sleep settings for CXPI bus signal triggering.

:TRIGger{[:ATRigger]|:BTRigger}:CXPI:WAKeupsleep:TYPE

Function Sets or queries the wakeup and sleep conditions for CXPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRi
gger}:CXPI:WAKeupsleep:TY
PE {SFRame|SLEEP|WAKeup|WPULse}
:TRIGger{[:ATRigger]|:BTRigger}:CXPI
:WAKeupsleep:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:CXPI:WAKEUPSLEEP:T
YPE SFRAME
:TRIGGER:ATRIGGER:CXPI:WAKEUPSLEEP:T
YPE?
-> :TRIGGER:ATRIGGER:CXPI:WAKEUPSLEE
P:TYPE SFRAME
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray?

Function Queries all FlexRay bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BCHannel

Function Sets or queries the channel bus type for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray:BCHannel {A|B}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray:BCHannel?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:BCHANNEL A
:TRIGGER:ATRIGGER:FLEXRAY:BCHANNEL?
-> :TRIGGER:ATRIGGER:FLEXRAY:BCHANN
EL A
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BRATe

Function Sets or queries the FlexRay bus signal trigger bit rate (data transfer rate).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BRATe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BRATe?
```
<NRf> = 2500000, 5000000, 10000000

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:BRATE 5000000
:TRIGGER:ATRIGGER:FLEXRAY:BRATE?
-> :TRIGGER:ATRIGGER:FLEXRAY:BRATE 5000000
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor?

Function Queries all FlexRay bus signal trigger error settings.

Syntax Queries all FlexRay bus signal trigger error settings.
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:BSS

Function Sets or queries the FlexRay bus signal trigger BSS error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:BSS {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:BSS?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:ERROR:BSS ON
:TRIGGER:ATRIGGER:FLEXRAY:ERROR:BSS?
-> :TRIGGER:ATRIGGER:FLEXRAY:ERROR:BSS 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:CRC

Function Sets or queries the FlexRay bus signal trigger CRC error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:CRC {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:CRC?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:ERROR:CRC ON
:TRIGGER:ATRIGGER:FLEXRAY:ERROR:CRC?
-> :TRIGGER:ATRIGGER:FLEXRAY:ERROR:CRC 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:FES

Function Sets or queries the FlexRay bus signal trigger FES error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:FES {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:FES?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:ERROR:FES ON
:TRIGGER:ATRIGGER:FLEXRAY:ERROR:FES?
-> :TRIGGER:ATRIGGER:FLEXRAY:ERROR:FES 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDData?

Function Queries all ID and data condition settings for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDData?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt?

Function Queries the cycle count, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition

Function Sets or queries the cycle count data condition, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:CONDITION EQUAL
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt<x>

Function Sets or queries the cycle count, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt<x>?
```
<x> = 1 or 2
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:COUNT1 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:COUNT1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:COUNT1 1
```

Description
- When :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDitionEQUal|GREater|NOTEqual is specified use :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt1 to set the value.
- When :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDitionLESS is specified, use :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt2 to set the value.
- When :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDitionBETWeen|NOTBetween is specified use :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt1 to set the larger value and :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt2 to set the smaller value.

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:MODE

Function Sets or queries the cycle count condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:MODE?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:MODE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:MODE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>?

Function Queries the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>?
```
<x> = 1 or 2

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:BCOunt

Function Sets or queries the pattern comparison position of the data field data, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:BCOunt {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:BCOunt?
```
<x> = 1 or 2
<NRf> = 0 to 253

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:BCOUNT 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:BCOUNT?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:BCOUNT 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:CONDition

Function Sets or queries the data field data condition, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:CONDITION EQUAL
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DBYTe

Function Sets or queries the number of bytes of the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DBYTe?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:DBYTE 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:DBYTE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:DBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DECimal<y>

Function Sets a reference value, which is one of the ID and data conditions, for FlexRay bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
FLEXray[:IDData]:DATA<x>:DECimal
<y> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:DATA<x>:DECimal<y>?
```
<x> = 1 or 2
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A1:DECIMAL1 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A1:DECIMAL1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
DATA1:DECIMAL1 1.000E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA:DPATtern:HEXa

Function Sets a reference value for the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:F
LEXray[:IDData]:DATA:DPATtern:HE
Xa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA:DPATtern:PATTern

Function Sets or queries the data field data, which is one of the ID and data conditions, for FlexRay bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FL
EXray[:IDData]:DATA:DPATtern:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:DATA:DPATtern:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A:DPATTERN:PATTERN "00110101"
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A:DPATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:ENDian

Function Sets or queries the data field endian, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:FLEXray[:IDData]:DATA<x>:ENDi
an {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:DATA<x>:ENDian?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A1:ENDIAN BIG
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A1:ENDIAN?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
DATA1:ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MODE

Function Sets or queries the data field condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger
}:FLEXray[:IDData]:DATA<x>:MO
DE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:DATA<x>:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A:MODE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A:MODE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MSBLsb

Function Sets or queries the MSB/LSB bit of the data field data, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:FLEXray[:IDData]:DATA<x>:MSBL
sb {<NRf>,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:DATA<x>:MSBLsb?
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A1:MSBLSB 7,0
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A1:MSBLSB?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
DATA1:MSBLSB 7,0
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:SIGN

Function Sets or queries the data sign of the data field, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger
}:FLEXray[:IDData]:DATA<x>:SI
GN {SIGN|UNSign}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:DATA<x>:SIGN?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A:SIGN SIGN
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DAT
A:SIGN?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
DATA:SIGN SIGN
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID?

Function Queries the frame ID, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:FID?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:CONDition

Function Sets or queries the frame ID data condition, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:FID:CONDition {EQUal|GR
Eater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:FID:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID
:CONDITION EQUAL
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID
:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
FID:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:ID<x>

Function Sets or queries the frame ID value, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:FID:ID<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:FID:ID<x>?
```
<x> = 1 or 2

<NRf> = 1 or 2047

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID
:ID1 100
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID
:ID1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
FID:ID1 100
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:MODE

Function Sets or queries the frame ID condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:FID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:FID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID
:MODE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID
:MODE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
FID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator?

Function Queries the indicator, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:INDIcator?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:MODE

Function Sets or queries the indicator condition (enabled/disabled), which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}
:FLEXray[:IDData]:INDIcator:MO
DE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:INDIcator:MODE?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:MODE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:MODE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
INDICATOR:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:NFRame

Function Sets or queries the indicator null frame, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
FLEXray[:IDData]:INDIcator:NFRa
me {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:INDIcator:NFRame?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:NFRAME 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:NFRAME?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
INDICATOR:NFRAME 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:PPReamble

Function Sets or queries the indicator payload preamble, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:F
LEXray[:IDData]:INDIcator:PPReamb
le {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:INDIcator:PPReamble?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:PPREAMBLE 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:PPREAMBLE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
INDICATOR:PPREAMBLE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:STFRame

Function Sets or queries the indicator start frame, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
FLEXray[:IDData]:INDIcator:STFRa
me {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:INDIcator:STFRame?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:STFRAME 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:STFRAME?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
INDICATOR:STFRAME 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:SYFRame

Function Sets or queries the indicator sync frame, which is one of the ID and data conditions, for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
FLEXray[:IDData]:INDIcator:SYFRa
me {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray[:IDData]:INDIcator:SYFRame?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:SYFRAME 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:IND
ICATOR:SYFRAME?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:
INDICATOR:SYFRAME 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR?

Function Queries all OR condition settings for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray:IDOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>?

Function Queries all settings related to the specified frame ID of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEX
ray:IDOR:ID<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt?

Function Queries all settings related to the cycle count of the specified frame ID of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:CONDition

Function Sets or queries the cycle count of the specified frame ID of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:CONDition {DONTcare|EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:CONDition?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CCOUNT:CONDITION EQUAL
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CCOUNT:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CCOUNT:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:COUNt<y>

Function Sets or queries the reference cycle count value of the specified frame ID of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:COUNt<y> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:COUNt<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CCOUNT:COUNT1 100
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CCOUNT:COUNT1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CCOUNT:COUNT1 100
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID?

Function Queries all settings related to the ID condition of the specified frame ID of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:CONDition

Function Sets or queries the ID judgment condition of the specified frame ID of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:CONDition?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FID:CONDITION EQUAL
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FID:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FID:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:ID<y>

Function Sets or queries the ID reference value of the specified frame ID of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:ID<y> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:ID<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 or 2047

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FID:ID1 1
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FID:ID1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FID:ID1 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:MODE

Function Sets or queries the specified frame ID condition (enabled/disabled) of the OR condition for FlexRay bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:MODE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:MODE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:MODE

Function Sets or queries the FlexRay bus signal trigger mode.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:MODE {ERRor|FSTart|IDData|IDOR}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:MODE?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:MODE FSTART
:TRIGGER:ATRIGGER:FLEXRAY:MODE?
-> :TRIGGER:ATRIGGER:FLEXRAY:MODE FSTART
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:SOURce

Function Sets or queries the FlexRay bus signal trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:SOURCE 1
:TRIGGER:ATRIGGER:FLEXRAY:SOURCE?
-> :TRIGGER:ATRIGGER:FLEXRAY:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C?

Function Queries all I2C bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess?

Function Queries all settings related to the address of the I2C bus signal trigger address pattern.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess?

Function Queries all settings related to the address of the I2C bus signal trigger address pattern.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress?

Function Queries all I2C bus signal trigger 10-bit address settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:DIRection

Function Sets or queries the 10-bit address data direction for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:DIRection {READ|WRITe|X}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:DIRection?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT10ADDRESS:DIRECTION READ
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT10ADDRESS:DIRECTION?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT10ADDRESS:DIRECTION READ
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:HEXa

Function Sets the 10-bit address for I2C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT10ADDRESS:HEXA "1AB"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:PATTern

Function Sets or queries the 10-bit address for I2C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C
:ADDRess:ADDRess:BIT10ADdress:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C
:ADDRess:ADDRess:BIT10ADdress:PATTe
rn?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT10ADDRESS:PATTERN "11010111001"
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT10ADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRE
SS:ADDRESS:BIT10ADDRESS:PATTE
RN "11010111001"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7ADdress?

Function Queries all I2C bus signal trigger 7-bit address settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT7ADdress?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:DIRection

Function Sets or queries the 7-bit address data direction for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess[:BIT7ADdress]:DIRect
ion {READ|WRITe|X}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess[:BIT7ADdress]:DIRect
ion?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT7ADDRESS:DIRECTION READ
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT7ADDRESS:DIRECTION?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADD
RESS:BIT7ADDRESS:DIRECTION READ
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:HEXa

Function Sets the 7-bit address for I2C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I
2C:ADDRess:ADDRess:BIT7ADdress:HE
Xa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT7ADDRESS:HEXA "5C"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:PATTern

Function Sets or queries the 7-bit address for I2C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess[:BIT7ADdress]:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess[:BIT7ADdress]:PATTe
rn?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT7ADDRESS:PATTERN "11100110"
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT7ADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADD
RESS:BIT7ADDRESS:PATTERN "11100110"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub?

Function Queries all I2C bus signal trigger 7-bit + sub address settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT7APsub?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit + sub address for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT7APsub:ADDRess?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:DIRection

Function Sets or queries the 7-bit + sub address data direction for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT7APsub:ADDRess:DI
Rection {READ|WRITe|X}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT7APsub:ADDRess:DI
Rection?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT7APSUB:ADDRESS:DIRECTION READ
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRES
S:BIT7APSUB:ADDRESS:DIRECTION?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7APSUB:ADDRESS:DIRECTI
ON READ
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:HEXa

Function Sets the 7-bit address of 7-bit + sub address for I2C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:HEXA "AB"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern

Function Sets or queries the 7-bit address of 7-bit + sub address for I2C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:PATTERN "00111010"
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:ADDRESS:PATTERN "00111010"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress?

Function Queries all settings related to the sub address of the 7-bit + sub address for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:HEXa

Function Sets the sub address of 7-bit + sub address for I2C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:HEXA "EF"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern

Function Sets or queries the sub address of 7-bit + sub address for I2C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:PATTERN "00111010"
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7APSUB:SADDRESS:PATTERN "00111010"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:MODE

Function Sets or queries the address condition enable/disable state for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:MODE?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:MODE ON
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:MODE?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:TYPE

Function Sets or queries the address condition address format for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:TYPE {BIT10ADdress|BIT7ADdress|BIT7APsub}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:TYPE BIT10ADDRESS
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:TYPE?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:TYPE BIT10ADDRESS
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA?

Function Queries all settings related to the data of the I2C bus signal trigger address pattern.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BCOunt

Function Sets or queries the data pattern comparison position for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BCOunt {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BCOunt?
```
<NRf> = 0 to 9999

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:BCOUNT 0
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:BCOUNT?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:BCOUNT 0
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BMODe

Function Sets or queries on/off state of the data pattern comparison position for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BMODe {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BMODe?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:BMODE ON
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:BMODE?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:BMODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:CONDition

Function Sets or queries the data judgment method (true/false) for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:CONDition {FALSe|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:CONDITION FALSE
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:CONDITION FALSE
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DBYTe

Function Sets or queries the number of setup data values for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DBYTe?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DBYTE 1
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DBYTE?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DPATtern:HEXa

Function Sets the data for I2C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DPATtern:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DPATTERN:HEXA "AB"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DPATtern:PATTern

Function Sets or queries the data condition data for I2C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DPATtern:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DPATtern:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DPATTERN:PATTERN "10001101"
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DPATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DPATTERN:PATTERN1 "10001101"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:MODE

Function Sets or queries the data condition enable/disable state for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:MODE ON
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:MODE?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall?

Function Queries all I2C bus signal trigger general call settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress?

Function Queries all settings related to the 7-bit master address of general call for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:HEXa

Function Sets the 7-bit master address of general call for I2C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:GENERALCALL:BIT7MADDRESS:HEXA "AB"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:PATTern

Function Sets or queries the 7-bit master address of general call for I2C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:GENERALCALL:BIT7MADDRESS:PATTERN "0010110"
:TRIGGER:ATRIGGER:I2C:GENERALCALL:BIT7MADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:GENERALCALL:BIT7MADDRESS:PATTERN "0010110"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:SBYTe

Function Sets or queries the general call second byte type for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:SBYTe {BIT7Maddress|DONTcare|H04|H06}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:SBYTe?
```

Example
```
:TRIGGER:ATRIGGER:I2C:GENERALCALL:SBYTE BIT7MADDRESS
:TRIGGER:ATRIGGER:I2C:GENERALCALL:SBYTE?
-> :TRIGGER:ATRIGGER:I2C:GENERALCALL:SBYTE BIT7MADDRESS
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:INCLuderw

Function Sets or queries the enable/disable condition of address R/W for I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:INCLuderw {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:INCLuderw?
```

Example
```
:TRIGGER:ATRIGGER:I2C:INCLUDERW ON
:TRIGGER:ATRIGGER:I2C:INCLUDERW?
-> :TRIGGER:ATRIGGER:I2C:INCLUDERW 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:MODE

Function Sets or queries the I2C bus signal trigger mode.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:MODE {ADRData|EVERystart|GENeralcall|HSMode|NONack|STARtbyte}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:MODE?
```

Example
```
:TRIGGER:ATRIGGER:I2C:MODE ADRDATA
:TRIGGER:ATRIGGER:I2C:MODE?
-> :TRIGGER:ATRIGGER:I2C:MODE ADRDATA
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack?

Function Queries all I2C bus signal trigger NON ACK ignore mode settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:HSMode

Function Sets or queries whether NONACK is to be ignored in high-speed mode of I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:HSMode {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:HSMode?
```

Example
```
:TRIGGER:ATRIGGER:I2C:NONACK:HSMODE ON
:TRIGGER:ATRIGGER:I2C:NONACK:HSMODE?
-> :TRIGGER:ATRIGGER:I2C:NONACK:HSMODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:READaccess

Function Sets or queries whether NON ACK is to be ignored in the read access byte of I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:READaccess {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:READaccess?
```

Example
```
:TRIGGER:ATRIGGER:I2C:NONACK:READACCESS ON
:TRIGGER:ATRIGGER:I2C:NONACK:READACCESS?
-> :TRIGGER:ATRIGGER:I2C:NONACK:READACCESS 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:STARtbyte

Function Sets or queries whether NON ACK is to be ignored in the start byte of I2C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:STARtbyte {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:STARtbyte?
```

Example
```
:TRIGGER:ATRIGGER:I2C:NONACK:STARTBYTE ON
:TRIGGER:ATRIGGER:I2C:NONACK:STARTBYTE?
-> :TRIGGER:ATRIGGER:I2C:NONACK:STARTBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:SCL

Function Sets or queries the I2C bus signal trigger clock signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:SCL {<NRf>|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:SCL?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:I2C:SCL 1
:TRIGGER:ATRIGGER:I2C:SCL?
-> :TRIGGER:ATRIGGER:I2C:SCL 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:SDA

Function Sets or queries the I2C bus signal trigger data signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:SDA {<NRf>|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:SDA?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:I2C:SDA 1
:TRIGGER:ATRIGGER:I2C:SDA?
-> :TRIGGER:ATRIGGER:I2C:SDA 1
```

Description The data signal that you can specify varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:I2C:SCL setting. For details, see the Features Guide.

:TRIGger{[:ATRigger]|:BTRigger}:LIN?

Function Queries all LIN bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:BLENgth

Function Sets or queries the LIN bus signal trigger break length.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:BLENgth {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:BLENgth?
```
<NRf> = 10 to 13

Example
```
:TRIGGER:ATRIGGER:LIN:BLENGTH 10
:TRIGGER:ATRIGGER:LIN:BLENGTH?
-> :TRIGGER:ATRIGGER:LIN:BLENGTH 10
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:BRATe

Function Sets or queries the LIN bus signal trigger bit rate (data transfer rate).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:BRATe {<NRf>|USER,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:BRATe?
```
<NRf> = 1200, 2400, 4800, 9600, 19200
USER <NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:LIN:BRATE 2400
:TRIGGER:ATRIGGER:LIN:BRATE?
-> :TRIGGER:ATRIGGER:LIN:BRATE 2400
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor?

Function Queries all LIN bus signal trigger error settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:PARity

Function Sets or queries the LIN bus signal trigger parity error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:PARity {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:PARity?
```

Example
```
:TRIGGER:ATRIGGER:LIN:ERROR:PARITY ON
:TRIGGER:ATRIGGER:LIN:ERROR:PARITY?
-> :TRIGGER:ATRIGGER:LIN:ERROR:PARITY 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:SYNCh

Function Sets or queries the LIN bus signal trigger synch error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:SYNCh {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:SYNCh?
```

Example
```
:TRIGGER:ATRIGGER:LIN:ERROR:SYNCH ON
:TRIGGER:ATRIGGER:LIN:ERROR:SYNCH?
-> :TRIGGER:ATRIGGER:LIN:ERROR:SYNCH 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData?

Function Queries all ID and data settings for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA?

Function Queries all LIN bus signal trigger data settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:CONDition

Function Sets or queries the LIN bus signal trigger data condition setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:CONDITION EQUAL
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DBYTe

Function Sets or queries the number of data bytes for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DBYTe?
```
<NRf> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DBYTE 1
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DBYTE?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DECimal<x>

Function Sets the data for LIN bus signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DECimal<x>?
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DECIMAL1 1
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DECIMAL1?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DECIMAL1 1.000E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DPATtern:HEXa

Function Sets the data for LIN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DPATtern:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DPATTERN:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DPATtern:PATTern

Function Sets or queries the data for LIN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DPATtern:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DPATtern:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DPATTERN:PATTERN "00110101"
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DPATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DPATTERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:ENDian

Function Sets or queries the data endian for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:ENDian {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:ENDian?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:ENDIAN BIG
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:ENDIAN?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MODE

Function Sets or queries the data condition enable/disable state for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MODE ON
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MODE?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MSBLsb

Function Sets or queries the MSB/LSB bit for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MSBLsb?
```
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MSBLSB 0
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MSBLSB?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MSBLSB 0
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:SIGN

Function Sets or queries the data sign for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:SIGN {SIGN|UNSign}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:SIGN?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:SIGN SIGN
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:SIGN?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:SIGN SIGN
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier?

Function Queries all LIN bus signal trigger identifier settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID?

Function Queries all LIN bus signal trigger ID settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:HEXa

Function Sets the ID for LIN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:HEXa {<String>}
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:HEXA "1E"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:MODE

Function Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:MODE ON
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:MODE?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:PATTern

Function Sets or queries the ID for LIN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:PATTERN "101100"
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:PATTERN?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:PATTERN "101100"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR?

Function Queries all ID OR condition settings for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier?

Function Queries all ID settings for the ID OR conditions for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>?

Function Queries all settings related to the specified ID of the ID OR condition for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:HEXa

Function Sets the ID value, which is one of the ID OR conditions, for LIN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:ID1:HEXA "1E"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID OR conditions, for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:ID1:MODE ON
:TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:ID1:MODE?
-> :TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:ID1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:PATTern

Function Sets or queries the ID condition, which is one of the ID OR conditions, for LIN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:PATTern?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:ID1:PATTERN "101100"
:TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:ID1:PATTERN?
-> :TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:ID1:PATTERN "101100"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:MODE

Function Sets or queries the ID condition enable/disable state for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:MODE?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:MODE ON
:TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:MODE?
-> :TRIGGER:ATRIGGER:LIN:IDOR:IDENTIFIER:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:MODE

Function Sets or queries the LIN bus signal trigger mode.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:MODE {BSYNch|ERRor|IDData|IDOR}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:MODE?
```

Example
```
:TRIGGER:ATRIGGER:LIN:MODE BSYNCH
:TRIGGER:ATRIGGER:LIN:MODE?
-> :TRIGGER:ATRIGGER:LIN:MODE BSYNCH
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:SOURce

Function Sets or queries the LIN bus signal trigger signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:LIN:SOURCE 1
:TRIGGER:ATRIGGER:LIN:SOURCE?
-> :TRIGGER:ATRIGGER:LIN:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:SPOint

Function Sets or queries the LIN bus signal trigger sample point.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:SPOint {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:SPOint?
```
<NRf> = 18.8 to 90.6

Example
```
:TRIGGER:ATRIGGER:LIN:SPOINT 18.8
:TRIGGER:ATRIGGER:LIN:SPOINT?
-> :TRIGGER:ATRIGGER:LIN:SPOINT 18.8
```

:TRIGger{[:ATRigger]|:BTRigger}:OR?

Function Queries all edge OR trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:OR?
```

:TRIGger{[:ATRigger]|:BTRigger}:OR:ALL

Function Sets or queries whether to make all channels edge OR trigger sources at once.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:OR:ALL {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:OR:ALL?
```

Example
```
:TRIGGER:ATRIGGER:OR:ALL ON
:TRIGGER:ATRIGGER:OR:ALL?
-> :TRIGGER:ATRIGGER:OR:ALL 1
```

:TRIGger{[:ATRigger]|:BTRigger}:OR:CHANnel<x>

Function Sets or queries the slope of each channel of the edge OR trigger.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:OR:CHANnel<x>{BOTH|FALL|OFF|RISE}
:TRIGger{[:ATRigger]|:BTRigger}:OR:CHANnel<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:OR:CHANNEL1 RISE
:TRIGGER:ATRIGGER:OR:CHANNEL1?
-> :TRIGGER:ATRIGGER:OR:CHANNEL1 RISE
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern?

Function Queries all settings related to the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern?
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS?

Function Queries all settings related to the logic inputs of the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS?
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:HEXa

Function Sets the logic input state of the combination (pattern) trigger of multiple inputs in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:BITS:HEXA "1F"
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:PATTern

Function Sets or queries the logic input pattern of the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:BITS:PATTERN "10111000"
:TRIGGER:ATRIGGER:PATTERN:BITS:PATTERN?
-> :TRIGGER:ATRIGGER:PATTERN:BITS:PATTERN "10111000"
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CHANnel<x>

Function Sets or queries the pattern of the specified channel for the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CHANnel<x> {DONTcare|HIGH|LOW}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CHANnel<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:PATTERN:CHANNEL1 DONTCARE
:TRIGGER:ATRIGGER:PATTERN:CHANNEL1?
-> :TRIGGER:ATRIGGER:PATTERN:CHANNEL1 DONTCARE
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk?

Function Queries all settings related to the clock signal for the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk?
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SLOPe

Function Sets or queries the clock signal slope for the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SLOPe {BOTH|FALL|RISE}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SLOPe?
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:CLOCK:SLOPE FALL
:TRIGGER:ATRIGGER:PATTERN:CLOCK:SLOPE?
-> :TRIGGER:ATRIGGER:PATTERN:CLOCK:SLOPE FALL
```

Description
- An error will occur if :TRIGger:PATTern:CLOCk:SOURce NONE is specified.
- If :TRIGger:SOURce:CHANnel<x>:WINDow is set to ON, select from {ENTer|EXIT}; otherwise, select from {FALL|RISE}.

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SOURce

Function Sets or queries the clock signal for the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SOURce {<NRf>|BIT<x>|NONE}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:PATTERN:CLOCK:SOURCE 1
:TRIGGER:ATRIGGER:PATTERN:CLOCK:SOURCE?
-> :TRIGGER:ATRIGGER:PATTERN:CLOCK:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CONDition

Function Sets or queries the trigger condition for the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CONDition {ENTer|EXIT|FALSe|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:CONDITION ENTER
:TRIGGER:ATRIGGER:PATTERN:CONDITION?
-> :TRIGGER:ATRIGGER:PATTERN:CONDITION ENTER
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:LOGic

Function Sets or queries the trigger combination condition for the combination (pattern) trigger of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:LOGic {AND|OR}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:LOGic?
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:LOGIC AND
:TRIGGER:ATRIGGER:PATTERN:LOGIC?
-> :TRIGGER:ATRIGGER:PATTERN:LOGIC AND
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:TQUalify

Function Sets or queries the time condition for combination (pattern) triggering of multiple inputs.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:TQUalify {INSide|LTHan|MTHan|OUTSide|TIMeout}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:TQUalify?
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:TQUALFY INSIDE
:TRIGGER:ATRIGGER:PATTERN:TQUALFY?
-> :TRIGGER:ATRIGGER:PATTERN:TQUALFY INSIDE
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT?

Function Queries all SENT signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:CEFactor?

Function Queries all SENT signal trigger error factor settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:CEFactor?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:CEFactor:SAComm?

Function Queries all status and communication error factor settings of SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:CEFactor:SAComm?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:CEFactor:SAComm:BIT<x>

Function Sets or queries status and communication error factor bit of SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CEFactor:SAComm:BIT<x> {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CEFactor:SAComm:BIT<x>?
```
<x> = 0, 1

Example
```
:TRIGGER:ATRIGGER:SENT:CEFACTOR:SACO
MM:BIT1 ON
:TRIGGER:ATRIGGER:SENT:CEFACTOR:SACO
MM:BIT1?
-> :TRIGGER:ATRIGGER:SENT:CEFACTOR:S
CPULSES 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:CEFactor:SCPulses

Function Sets or queries the consecutive calibration pulse error factor of SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRi
gger}:SENT:CEFactor:SCPuls
es {OFF|OPT2|POPTion}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CEFactor:SCPulses?
```

Example
```
:TRIGGER:ATRIGGER:SENT:CEFACTOR:SCPU
LSES OFF
:TRIGGER:ATRIGGER:SENT:CEFACTOR:SCPU
LSES?
-> :TRIGGER:ATRIGGER:SENT:CEFACTOR:S
CPULSES OFF
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:CRCType

Function Sets or queries the SENT signal trigger CRC computation type.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CRCType {LEGacy|RECommended}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CRCType?
```

Example
```
:TRIGGER:ATRIGGER:SENT:CRCTYPE LEGAC
Y
:TRIGGER:ATRIGGER:SENT:CRCTYPE?
-> :TRIGGER:ATRIGGER:SENT:CRCTYPE LE
GACY
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:CTICk

Function Sets or queries the SENT signal trigger clock tick value.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CTICk {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CTICk?
```
<NRf> = 1.00us to 100.00us (resolution: 0.01us)

Example
```
:TRIGGER:ATRIGGER:SENT:CTICK 0.00000
1
:TRIGGER:ATRIGGER:SENT:CTICK?
-> :TRIGGER:ATRIGGER:SENT:
CTICK 1.000000E-06
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:CTOLerance

Function Sets or queries the SENT signal trigger clock tolerance.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CTOLerance {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:CTOLerance?
```
<NRf> = ±1.0% to ±30.0% (resolution: 0.1%)

Example
```
:TRIGGER:ATRIGGER:SENT:CTOLERANCE 25
.0
:TRIGGER:ATRIGGER:SENT:CTOLERANCE?
-> :TRIGGER:ATRIGGER:SENT:CTOLERANC
E 25.0E+00
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:DNIBbles

Function Sets or queries the number of data nibbles for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:DNIBbles {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:DNIBbles?
```
<NRf> = 1 to 6

Example
```
:TRIGGER:ATRIGGER:SENT:DNIBBLES 6
:TRIGGER:ATRIGGER:SENT:DNIBBLES?
-> :TRIGGER:ATRIGGER:SENT:DNIBBLES 6
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:DTYPe

Function Sets or queries the SENT signal trigger fast channel data format.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:DTYPe {NIBBle|USER}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:DTYPe?
```

Example
```
:TRIGGER:ATRIGGER:SENT:DTYPE NIBBLE
:TRIGGER:ATRIGGER:SENT:DTYPE?
-> :TRIGGER:ATRIGGER:SENT:DTYPE NIBB
LE
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:ERRor?

Function Queries all SENT signal trigger error settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:ERRor:FCRC

Function Sets or queries the SENT signal trigger fast channel CRC error.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:FCRC {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:FCRC?
```

Example
```
:TRIGGER:ATRIGGER:SENT:ERROR:FCRC ON
:TRIGGER:ATRIGGER:SENT:ERROR:FCRC?
-> :TRIGGER:ATRIGGER:SENT:ERROR:FCR
C 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:ERRor:NDValue

Function Sets or queries the SENT signal trigger nibble data value error.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:NDValue {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:NDValue?
```

Example
```
:TRIGGER:ATRIGGER:SENT:ERROR:NDVAL
UE ON
:TRIGGER:ATRIGGER:SENT:ERROR:NDVAL
UE?
-> :TRIGGER:ATRIGGER:SENT:ERROR:NDVA
LUE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:ERRor:NNUMber

Function Sets or queries the SENT signal trigger nibble data count error.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:NNUMber {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:NNUMber?
```

Example
```
:TRIGGER:ATRIGGER:SENT:ERROR:NNUMB
ER ON
:TRIGGER:ATRIGGER:SENT:ERROR:NNUMB
ER?
-> :TRIGGER:ATRIGGER:SENT:ERROR:NNUM
BER 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:ERRor:SAComm

Function Sets or queries the SENT signal trigger status and communication error.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:SAComm {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:SAComm?
```

Example
```
:TRIGGER:ATRIGGER:SENT:ERROR:SACO
MM ON
:TRIGGER:ATRIGGER:SENT:ERROR:SACOMM?
-> :TRIGGER:ATRIGGER:SENT:ERROR:SACO
MM 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:ERRor:SCRC

Function Sets or queries the SENT signal trigger slow channel CRC error.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:SCRC {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:SCRC?
```

Example
```
:TRIGGER:ATRIGGER:SENT:ERROR:SCRC ON
:TRIGGER:ATRIGGER:SENT:ERROR:SCRC?
-> :TRIGGER:ATRIGGER:SENT:ERROR:SCR
C 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:ERRor:SCPulses

Function Sets or queries the consecutive calibration pulse error for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:SCPulses {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:ERRor:SCPulses?
```

Example
```
:TRIGGER:ATRIGGER:SENT:ERROR:SCPULS
ES ON
:TRIGger:ATRIGGER:SENT:ERROR:SCPULS
ES?
-> :TRIGGER:ATRIGGER:SENT:ERROR:SCPU
LSES 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa?

Function Queries all fast channel data settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DATA<x>?

Function Queries all settings related to the specified fast channel user data for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DATA<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DATA<x>:CONDition

Function Sets or queries the comparison condition of the specified user data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DATA<x>:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DATA<x>:CONDition?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SENT:FDATA:DATA1:CONDITION EQUAL
:TRIGGER:ATRIGGER:SENT:FDATA:DATA1:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:FDATA:DATA1:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DATA<x>:DECimal<y>

Function Sets or queries the data of the specified user data of the SENT signal trigger fast channel in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DATA<x>:DECimal<y> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DATA<x>:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:SENT:FDATA:DATA1:DECIMAL1 1
:TRIGGER:ATRIGGER:SENT:FDATA:DATA1:DECIMAL1?
-> :TRIGGER:ATRIGGER:SENT:FDATA:DATA1:DECIMAL1 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DATA<x>:MODE

Function Sets or queries the data enable/disable condition of the specified user data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DATA<x>:MODE{<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DATA<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SENT:FDATA:DATA1:MODE ON
:TRIGGER:ATRIGGER:SENT:FDATA:DATA1:MODE?
-> :TRIGGER:ATRIGGER:SENT:FDATA:DATA1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DNIBbles?

Function Queries all fast channel nibble data settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DNIBbles?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DNIBbles:CONDition

Function Sets or queries the nibble data comparison condition of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DNIBbles:CONDition {FALSe|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DNIBbles:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:SENT:FDATA:DNIBBLES:CONDITION TRUE
:TRIGGER:ATRIGGER:SENT:FDATA:DNIBBLES:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:FDATA:DNIBBLES:CONDITION TRUE
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DNIBbles:HEXa

Function Sets the SENT signal trigger fast channel nibble data in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DNIBbles:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:SENT:FDATA:DNIBBLES:HEXA "112233"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FDATa:DNIBbles:PATTern

Function Sets or queries the SENT signal trigger fast channel nibble data in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DNIBbles:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FDATa:DNIBbles:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:SENT:FDATA:DNIBBLE
S:PATTERN "111100001111000011110000"
:TRIGGER:ATRIGGER:SENT:FDATA:DNIBBLE
S:PATTERN?
-> :TRIGGER:ATRIGGER:SE
NT:FDATA:DNIBBLES:PATTE
RN "111100001111000011110000"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FSAComm?

Function Queries all fast channel status and communication nibble settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FSAComm?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FSAComm:HEXa

Function Sets the SENT signal trigger fast channel status and communication nibble data in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FSAComm:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:SENT:FSACOMM:HE
XA "F"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:FSAComm:PATTern

Function Sets or queries the SENT signal trigger fast channel status and communication nibble data in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FSAComm:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:FSAComm:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:SENT:FSACOMM:PATTE
RN "1111"
:TRIGGER:ATRIGGER:SENT:FSACOMM:PATTE
RN?
-> :TRIGGER:ATRIGGER:SENT:FSACOMM:PA
TTERN "1111"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:MODE

Function Sets or queries the SENT signal trigger mode.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:MODE {EFASt|ERRor|ESLow|FDATa|FSACo
mm|SDATa}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SENT:MODE ERRor
:TRIGGER:ATRIGGER:SENT:MODE?
-> :TRIGGER:ATRIGGER:SENT:MODE ERRor
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:PPULse

Function Sets or queries the presence or absence of pause pulses for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:PPULse {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:PPULse?
```

Example
```
:TRIGGER:ATRIGGER:SENT:PPULSE ON
:TRIGger:ATRIGGER:SENT:PPULSE?
-> :TRIGGER:ATRIGGER:SENT:PPULSE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa?

Function Queries all slow channel data settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced?

Function Queries all slow channel enhanced type settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:CBIT

Function Sets or queries the SENT signal trigger slow channel enhanced type configuration bit.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:CBIT {D12Bit|D16Bit}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:CBIT?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCE
D:CBIT D12Bit
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCE
D:CBIT?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHA
NCED:CBIT D12Bit
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit?

Function Queries all slow channel enhanced type 12-bit data and 8-bit ID settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:DATA?

Function Queries all slow channel enhanced type 12-bit data and 8-bit ID data settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:DATA:CONDition

Function Sets or queries the data comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit:DATA:CONDiti
on {EQUal|FALSE|GREater|INSide|LESS|
NOTEqual|OUTSide|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit:DATA:CONDiti
on?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:CONDITION EQUAL
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHA
NCED:D12BIT:DATA:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:DATA:DECimal<x>

Function Sets or queries the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit:DATA:DECimal
<x> {<NRf>}
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:DECIMAL1 123
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:DECIMAL1?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHA
NCED:D12BIT:DATA:DECIMAL1 123
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:DATA:HEXa

Function Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:S
ENT:SDATa:ENHanced:D12Bit:DATA:HE
Xa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:HEXA "123"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:DATA:MODE

Function Sets or queries the data enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:S
ENT:SDATa:ENHanced:D12Bit:DATA:MO
DE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:MODE ON
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:MODE?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHA
NCED:D12BIT:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:DATA:PATTern

Function Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SEN
T:SDATa:ENHanced:D12Bit:DATA:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit:DATA:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:PATTERN "111101010101"
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCE
D:D12BIT:DATA:PATTERN?
-> :TRIGGER:ATRIGGER:SENT:SDA
TA:ENHANCED:D12BIT:DATA:PATTE
RN "111101010101"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID?

Function Queries all ID settings related to the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:ENHanced:D12Bit:ID?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:CONDition

Function Sets or queries the ID comparison condition of slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:CONDITION EQUAL
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:DECimal<x>

Function Sets the ID of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:HEXa<x>?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:DECIMAL1 0
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:DECIMAL1?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:DECIMAL1 0
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:MODE

Function Sets or queries the ID enable/disable condition of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D12Bit:ID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:MODE ON
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:MODE?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D12BIT:ID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit?

Function Queries all slow channel enhanced type 16-bit data and 4-bit ID settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA?

Function Queries all slow channel enhanced type 16-bit data and 4-bit ID data settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:CONDition

Function Sets or queries the data comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:CONDition {EQUal|FALSE|GREater|INSide|LESS|NOTEqual|OUTSide|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:CONDITION EQUAL
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:DECimal<x>

Function Sets or queries the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:DECimal<x> {<NRf>}
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:DECIMAL 123
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:DECIMAL?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:DECIMAL 123
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:HEXa

Function Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:HEXA "0000"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:MODE

Function Sets or queries the data enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:MODE ON
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:MODE?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:PATTern

Function Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:DATA:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:PATTERN "1111000011110000"
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:PATTERN?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:DATA:PATTERN "1111000011110000"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID?

Function Queries all ID settings related to the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:CONDition

Function Sets or queries the ID comparison condition of slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:CONDition {EQUal|GREater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:CONDITION EQUAL
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:DECimal<x>

Function Sets the ID of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:DECimal<x>?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:DECIMAL1 0
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:DECIMAL1?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:DECIMAL1 0
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:MODE

Function Sets or queries the ID enable/disable condition of the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:ENHanced:D16Bit:ID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:MODE ON
:TRIGger:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:MODE?
-> :TRIGGER:ATRIGGER:SENT:SDATA:ENHANCED:D16BIT:ID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt?

Function Queries all slow channel short type settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:DATA?

Function Queries all slow channel short type data settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:DATA:CONDition

Function Sets or queries the short type data comparison condition of the SENT signal trigger slow channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:CONDition {EQUal|F
ALSE|GREater|INSide|LESS|NOTEqual|OU
TSide|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORT:D
ATA:CONDITION EQUAL
:TRIGger:ATRIGGER:SENT:SDATA:SHORT:D
ATA:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:SDATA:SHOR
T:DATA:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:DATA:DECimal<x>

Function Sets or queries the SENT signal trigger slow channel short type data in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:DECimal<x> {<NRf>}
```
<x> = 1 or 2
<NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORt:D
ATA:DECIMAL1 123
:TRIGGER:ATRIGGER:SENT:SDATA:SHORt:D
ATA:DECIMAL1?
-> :TRIGGER:ATRIGGER:SENT:SDATA:SHOR
t:DATA:DECIMAL1 123
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:DATA:HEXa

Function Sets the SENT signal trigger slow channel short type data in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORT:D
ATA:HEXA "00"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:DATA:MODE

Function Sets or queries the short type data enable/disable condition of the SENT signal trigger slow channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORT:D
ATA:MODE ON
:TRIGger:ATRIGGER:SENT:SDATA:SHORT:D
ATA:MODE?
-> :TRIGGER:ATRIGGER:SENT:SDATA:SHOR
T:DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:DATA:PATTern

Function Sets the SENT signal trigger slow channel short type data in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:DATA:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORT:D
ATA:PATTERN "11110000"
:TRIGger:ATRIGGER:SENT:SDATA:SHORT:D
ATA:PATTERN?
-> :TRIGGER:ATRIGGER:SENT:SDATA:SHOR
T:DATA:PATTERN "11110000"
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:ID?

Function Queries all slow channel short type ID settings for SENT signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:ID?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:ID:CONDition

Function Sets or queries the short type ID comparison condition of the SENT signal trigger slow channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:ID:CONDition {EQUal|GRE
ater|INSide|LESS|NOTEqual|OUTSide}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:ID:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORT:I
D:CONDITION EQUAL
:TRIGger:ATRIGGER:SENT:SDATA:SHORT:I
D:CONDITION?
-> :TRIGGER:ATRIGGER:SENT:SDATA:SHOR
T:ID:CONDITION EQUAL
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:ID:DECimal<x>

Function Sets the SENT signal trigger slow channel short type ID in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:ID:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:ID:DECimal<x>?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORT:I
D:DECimal1 0
:TRIGger:ATRIGGER:SENT:SDATA:SHORT:I
D:DECIMAL1?
-> :TRIGGER:ATRIGGER:SENT:SDATA:SHOR
T:ID:DECIMAL1 0
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SDATa:SHORt:ID:MODE

Function Sets or queries the short type ID enable/disable condition of the SENT signal trigger slow channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:ID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SDATa:SHORt:ID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SENT:SDATA:SHORT:I
D:MODE ON
:TRIGger:ATRIGGER:SENT:SDATA:SHORT:I
D:MODE?
-> :TRIGGER:ATRIGGER:SENT:SDATA:SHOR
T:ID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:SOURce

Function Sets or queries the SENT signal trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SOURce {<NRf>|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:SENT:SOURCE 1
:TRIGger:ATRIGGER:SENT:SOURCE?
-> :TRIGGER:ATRIGGER:SENT:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:STYPe

Function Sets or queries the SENT signal trigger slow channel format.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:STYPe {ENHanced|SHORt}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:STYPe?
```

Example
```
:TRIGGER:ATRIGGER:SENT:STYPE ENHANCE
D
:TRIGger:ATRIGGER:SENT:STYPE?
-> :TRIGGER:ATRIGGER:SENT:STYPE ENHA
NCED
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:USETup?

Function Queries all settings related to user-defined data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup?
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:USETup:DATA<x>?

Function Queries all settings related to user-defined data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:DATA<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:SENT:USETup:DATA<x>:MODE

Function Sets or queries the enable/disable condition of the specified user-defined data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:DATA<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:DATA<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SENT:USETUP:DATA1:
MODE ON
:TRIGger:ATRIGGER:SENT:USETUP:DATA1:
MODE?
-> :TRIGGER:ATRIGGER:SENT:USETUP:DAT
A1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:USETup:DATA<x>:ORDer

Function Sets or queries the byte order of the specified user-defined data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:DATA<x>:ORDer {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:DATA<x>:ORDer?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SENT:USETUP:DATA1:
ORDER BIG
:TRIGger:ATRIGGER:SENT:USETUP:DATA1:
ORDER?
-> :TRIGGER:ATRIGGER:SENT:USETUP:DAT
A1:ORDER BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:USETup:DATA<x>:SIZE

Function Sets or queries the data size of the specified user-defined data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:DATA<x>:SIZE {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:DATA<x>:SIZE?
```
<x> = 1 to 4
<NRf> = 0 to 24

Example
```
:TRIGGER:ATRIGGER:SENT:USETUP:DATA1:
SIZE 24
:TRIGger:ATRIGGER:SENT:USETUP:DATA1:
SIZE?
-> :TRIGGER:ATRIGGER:SENT:USETUP:DAT
A1:SIZE 24
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:USETup:MULTiplexing

Function Sets or queries the multiplexing enable/disable condition of the specified user-defined data of the SENT signal trigger fast channel.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:MULTiplexing {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:USETup:MULTiplexing?
```

Example
```
:TRIGGER:ATRIGGER:SENT:USETUP:MULTIP
LEXING ON
:TRIGGER:ATRIGGER:SENT:USETUP:MULTIP
LEXING?
-> :TRIGGER:ATRIGGER:SENT:USETUP:MUL
TIPLEXING ON
```

:TRIGger{[:ATRigger]|:BTRigger}:SENT:VERSion

Function Sets or queries the SENT signal trigger specification version.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:VERSion {APR2016|FEB2008|JAN2010}
:TRIGger{[:ATRigger]|:BTRigger}:SENT
:VERESion?
```

Example
```
:TRIGGER:ATRIGGER:SENT:VERSION JAN20
10
:TRIGger:ATRIGGER:SENT:VERSION?
-> :TRIGGER:ATRIGGER:SENT:VERSION JA
N2010
```

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle?

Function Queries all edge trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMP
le?
```

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:HFRejection(HighFrequencyREJECTION)

Function Sets or queries the on/off state of the low-pass filter (HF rejection) for the edge trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMP
le:HFRejection {<Frequency>|OFF}
:TRIGger{[:ATRigger]|:BTRigger}:SIMP
le:HFRejection?
```
<Frequency> = 15KHz to 20MHz

Example
```
:TRIGGER:ATRIGGER:SIMPLE:HFREJECTI
ON 15KHZ
:TRIGGER:ATRIGGER:SIMPLE:HFREJECTI
ON?
-> :TRIGGER:ATRIGGER:SIMPLE:HFREJECT
ION 15000
```

Description You cannot use this command or query if :TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce is set to EXTernal or LINE.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:LEVel

Function Sets or queries the trigger level of the edge trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:LEVel {<Voltage>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:LEVel?
```
<Voltage> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:SIMPLE:LEVEL 0V
:TRIGGER:ATRIGGER:SIMPLE:LEVEL?
-> :TRIGGER:ATRIGGER:SIMPLE:LEVEL 0.000E+00
```

Description
- You cannot use this command or query if :TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce is set to LINE.
- When a current probe is in use, this command does not set or query the <Current> value.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:NREJection

Function Sets or queries the noise rejection of the edge trigger source level.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:NREJection {HIGH|LOW|MIDDle}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:NREJection?
```

Example
```
:TRIGGER:ATRIGGER:SIMPLE:NREJECTION HIGH
:TRIGGER:ATRIGGER:SIMPLE:NREJECTION?
-> :TRIGGER:ATRIGGER:SIMPLE:NREJECTION HIGH
```

Description You cannot use this command or query if :TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce is set to EXTernal or LINE.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:PROBe

Function Sets or queries the probe of the external edge trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:PROBe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:PROBe?
```
<NRf> = 1, 10

Example
```
:TRIGGER:ATRIGGER:SIMPLE:PROBE 1
:TRIGGER:ATRIGGER:SIMPLE:PROBE?
-> :TRIGGER:ATRIGGER:SIMPLE:PROBE 1
```

Description You cannot use this command or query if :TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce is not set to EXTernal.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:RANGe

Function Sets or queries the input range of the external edge trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:RANGe {<Voltage>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:RANGe?
```
<Voltage> = 2 to 200 (V)

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SLOPe

Function Sets or queries the trigger slope (polarity if windows is ON) of the edge trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SLOPe {BOTH|FALL|RISE}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SLOPe?
```

Example
```
:TRIGGER:ATRIGGER:SIMPLE:SLOPE BOTH
:TRIGGER:ATRIGGER:SIMPLE:SLOPE?
-> :TRIGGER:ATRIGGER:SIMPLE:SLOPE BOTH
```

Description You cannot use this command or query if :TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce is set to LINE.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce

Function Sets or queries the edge trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce {<NRf>|BIT<x>|EXTernal|LINE}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:SIMPLE:SOURCE 1
:TRIGGER:ATRIGGER:SIMPLE:SOURCE?
-> :TRIGGER:ATRIGGER:SIMPLE:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern?

Function Queries all user-defined bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:BITSize

Function Sets or queries the user-defined bus signal trigger bit length.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:BITSize {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:BITSize?
```
<NRf> = 1 to 128

Example
```
:TRIGGER:ATRIGGER:SPATTERN:BITSIZE 1
:TRIGGER:ATRIGGER:SPATTERN:BITSIZE?
-> :TRIGGER:ATRIGGER:SPATTERN:BITSIZE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:BRATe

Function Sets or queries the user-defined bus signal trigger bit rate.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:BRATe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:BRATe?
```
<NRf> = 1000 to 50000000

Example
```
:TRIGGER:ATRIGGER:SPATTERN:BRATE 1000
:TRIGGER:ATRIGGER:SPATTERN:BRATE?
-> :TRIGGER:ATRIGGER:SPATTERN:BRATE 1000
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk?

Function Queries all user-defined bus signal trigger clock signal settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:MODE

Function Sets or queries the enable/disable condition of the user-defined bus signal trigger clock signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:MODE ON
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:MODE?
-> :TRIGGER:ATRIGGER:SPATTERN:CLOCK:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:POLarity

Function Sets or queries the polarity of the user-defined bus signal trigger clock signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:POLarity {FALL|RISE}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:POLARITY FALL
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:POLARITY?
-> :TRIGGER:ATRIGGER:SPATTERN:CLOCK:POLARITY FALL
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:SOURce

Function Sets or queries the user-defined bus signal trigger clock signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:CLOCK:SOURCE 1
```

Description The clock signal that you can specify varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce setting. For details, see the Features Guide.

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS?

Function Queries all user-defined bus signal trigger chip select signal settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:ACTive

Function Sets or queries the active state of the user-defined bus signal trigger chip select signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:ACTive {HIGH|LOW}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:ACTive?
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:CS:ACTIVE HIGH
:TRIGGER:ATRIGGER:SPATTERN:CS:ACTIVE?
-> :TRIGGER:ATRIGGER:SPATTERN:CS:ACTIVE HIGH
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:SOURce

Function Sets or queries the user-defined bus signal trigger chip select signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:SOURce {<NRf>|NONE}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SPATTERN:CS:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:CS:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:CS:SOURCE 1
```

Description The chip select signal that you can specify varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce setting. For details, see the Features Guide.

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA?

Function Queries all user-defined bus signal trigger data signal settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:ACTive

Function Sets or queries the active state of the user-defined bus signal trigger data signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:ACTive {HIGH|LOW}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:ACTive?
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:DATA:ACTIVE HIGH
:TRIGGER:ATRIGGER:SPATTERN:DATA:ACTIVE?
-> :TRIGGER:ATRIGGER:SPATTERN:DATA:ACTIVE HIGH
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce

Function Sets or queries the user-defined bus signal trigger data signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SPATTERN:DATA:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:DATA:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:DATA:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:HEXa

Function Queries the data conditions of the user-defined bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:HEXa {<String>}
:TRIGGER:ATRIGGER:SPATTERN:HEXA "12"
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh?

Function Queries all user-defined bus signal trigger latch signal settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:POLarity

Function Sets or queries the polarity of the user-defined bus signal trigger latch signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:POLarity {FALL|RISE}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:LATCH:POLARITY FALL
:TRIGGER:ATRIGGER:SPATTERN:LATCH:POLARITY?
-> :TRIGGER:ATRIGGER:SPATTERN:LATCH:POLARITY FALL
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:SOURce

Function Sets or queries the user-defined bus signal trigger latch signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:SOURce {<NRf>|NONE}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SPATTERN:LATCH:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:LATCH:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:LATCH:SOURCE 1
```

Description The latch signal that you can specify varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce setting. For details, see the Features Guide.

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:PATTern

Function Sets or queries the data conditions of the user-defined bus signal trigger in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:PATTERN "00110101"
:TRIGGER:ATRIGGER:SPATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:SPATTERN:PATTERN "00110101"
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI?

Function Queries all SPI bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:BITorder

Function Sets or queries the bit order of the SPI bus signal trigger data.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:BITorder {LSBFirst|MSBFirst}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:BITorder?
```

Example
```
:TRIGGER:ATRIGGER:SPI:BITORDER LSBFIRST
:TRIGGER:ATRIGGER:SPI:BITORDER?
-> :TRIGGER:ATRIGGER:SPI:BITORDER LSBFIRST
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk?

Function Queries all SPI bus signal trigger clock signal settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:POLarity

Function Sets or queries the polarity of the SPI bus signal trigger clock signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:POLarity {FALL|RISE}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:SPI:CLOCK:POLARITY FALL
:TRIGGER:ATRIGGER:SPI:CLOCK:POLARITY?
-> :TRIGGER:ATRIGGER:SPI:CLOCK:POLARITY FALL
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:SOURce

Function Sets or queries the SPI bus signal trigger clock signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:SOURce {<NRf>|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:SPI:CLOCK:SOURCE 1
:TRIGGER:ATRIGGER:SPI:CLOCK:SOURCE?
-> :TRIGGER:ATRIGGER:SPI:CLOCK:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CS?

Function Queries all SPI bus signal trigger chip select signal settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
CS?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CS:ACTive

Function Sets or queries the active state of the SPI bus signal trigger chip select signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
CS:ACTive {HIGH|LOW}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
CS:ACTive?
```

Example
```
:TRIGGER:ATRIGGER:SPI:CS:ACTIVE HIGH
:TRIGGER:ATRIGGER:SPI:CS:ACTIVE?
-> :TRIGGER:ATRIGGER:SPI:CS:ACTIVE H
IGH
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CS:SOURce

Function Sets or queries the SPI bus signal trigger chip select signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
CS:SOURce {<NRf>|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
CS:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:SPI:CS:SOURCE 1
:TRIGGER:ATRIGGER:SPI:CS:SOURCE?
-> :TRIGGER:ATRIGGER:SPI:CS:SOURCE 1
```

Description The chip select signal that you can specify varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:SOURce setting. For details, see the Features Guide.

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>?

Function Queries all SPI bus signal trigger data signal settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
DATA<x>?
```
<x> = 1 or 2

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:BCOunt

Function Sets or queries the pattern comparison start position of the specified data for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
DATA<x>:BCOunt {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
DATA<x>:BCOunt?
```
<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:BCOUNT 0
:TRIGGER:ATRIGGER:SPI:DATA1:BCOUNT?
-> :TRIGGER:ATRIGGER:SPI:DATA1:BCOU
NT 0
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:CONDition

Function Sets or queries the judgment method (true/false) of the specified data for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
DATA<x>:CONDition {FALSe|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
DATA<x>:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:CONDITI
ON FALSE
:TRIGGER:ATRIGGER:SPI:DATA1:CONDITI
ON?
-> :TRIGGER:ATRIGGER:SPI:DATA1:CONDI
TION FALSE
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DBYTe

Function Sets or queries the data size (number of bytes) of the specified data for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
DATA<x>:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:
DATA<x>:DBYTe?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:DBYTE 1
:TRIGGER:ATRIGGER:SPI:DATA1:DBYTE?
-> :TRIGGER:ATRIGGER:SPI:DATA1:DBYT
E 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DPATtern:HEXa

Function Sets the specified data for SPI bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DPATtern:HEXa {<String>}
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:DPATTERN:HEXA "AB"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DPATtern:PATTern

Function Sets or queries the specified data for SPI bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DPATtern:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DPATtern:PATTern?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:DPATTERN:PATTERN "11001010"
:TRIGGER:ATRIGGER:SPI:DATA1:DPATTERN:PATTERN?
-> :TRIGGER:ATRIGGER:SPI:DATA1:DPATTERN:PATTERN "11001010"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DBYTe setting.

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:MODE

Function Sets or queries the enable/disable condition of data signal for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:MODE ON
:TRIGGER:ATRIGGER:SPI:DATA1:MODE?
-> :TRIGGER:ATRIGGER:SPI:DATA1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:SOURce

Function Sets or queries the SPI bus signal trigger data signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:SOURce {<NRf>|BIT<y>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 4
<y> of BIT<y> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:SOURCE 1
:TRIGGER:ATRIGGER:SPI:DATA1:SOURCE?
-> :TRIGGER:ATRIGGER:SPI:DATA1:SOURCE 1
```

Description The data signal that you can specify varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:SOURce setting. For details, see the Features Guide.

:TRIGger{[:ATRigger]|:BTRigger}:SPI:MODE

Function Sets or queries the wiring system (three-wire system or four-wire system) of the SPI bus trigger.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:MODE {WIRE3|WIRE4}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:MODE?
```

Example
```
:TRIGGER:ATRIGGER:SPI:MODE WIRE3
:TRIGGER:ATRIGGER:SPI:MODE?
-> :TRIGGER:ATRIGGER:SPI:MODE WIRE3
```

:TRIGger{[:ATRigger]|:BTRigger}:TYPE

Function Sets or queries the trigger type.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:TYPE {INTerval|OR|PATTern|PULSe|RFTime|RUNT|SIMPle|TIMeout|TV|WINDow|WOR|CANBus|CANFDbus|CXPIbus|FLEXray|I2CBus|LINBus|SENT|SPATtern|SPIBus|UART}
:TRIGger{[:ATRigger]|:BTRigger}:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:TYPE SIMPLE
:TRIGGER:ATRIGGER:TYPE?
-> :TRIGGER:ATRIGGER:TYPE SIMPLE
```

Description The settings that are available for :TRIGger:BTRigger:TYPE are {OR|PATTern|SIMPle|WINDow|WOR|CANBus|CANFDbus|CXPIbus|FLEXray|I2CBus|LINBus|SENT|SPATtern|SPIBus|UART}.

:TRIGger{[:ATRigger]|:BTRigger}:UART?

Function Queries all UART signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}UART?
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:BITorder

Function Sets or queries the UART signal trigger bit order.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:BITorder {LSBFirst|MSBFirst}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:BITorder?
```

Example
```
:TRIGGER:ATRIGGER:UART:BITORDER LSBF
IRST
:TRIGGER:ATRIGGER:UART:BITORDER?
-> :TRIGGER:ATRIGGER:UART:BITORDER L
SBFIRST
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:BRATe

Function Sets or queries the UART signal trigger bit rate (data transfer rate).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:BRATe {<NRf>|USER,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:BRATe?
```
<NRf> = 1200, 2400, 4800, 9600, 19200
USER <NRf> = See the Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:UART:BRATE 2400
:TRIGGER:ATRIGGER:UART:BRATE?
-> :TRIGGER:ATRIGGER:UART:BRATE 2400
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA?

Function Queries all UART signal trigger data settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:ASCii

Function Sets the UART signal trigger data in ASCII format.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:ASCii {<String>}
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:ASCII "T
EST"
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:CONDition

Function Sets or queries the data judgment method (true/false) for UART signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:CONDition {DONTcare|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:CONDITI
ON DONTCARE
:TRIGGER:ATRIGGER:UART:DATA:CONDITI
ON?
-> :TRIGGER:ATRIGGER:UART:DATA:CONDI
TION DONTCARE
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:CSENsitive

Function Sets or queries whether to distinguish uppercase and lowercase letters in ASCII data for UART signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:CSENsitive {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:CSENsitive?
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:CSENSITI
VE ON
:TRIGGER:ATRIGGER:UART:DATA:CSENSITI
VE?
-> :TRIGGER:ATRIGGER:UART:DATA:CSENS
ITIVE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:DBYTe

Function Sets or queries the number of data bytes for UART signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:DBYTe?
```
<NRf> = 1 to 4 (byte)

Example
```
:TRIGGER:ATRIGGER:UART:DATA:DBYTE 1
:TRIGGER:ATRIGGER:UART:DATA:DBYTE?
-> :TRIGGER:ATRIGGER:UART:DATA:DBYT
E 1
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:DPATtern:HEXa

Function Sets the data for UART signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:DPATtern:HEXa {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:DPATtern:HEXa?
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:DPATTERN
:HEXA "12"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:DBYTe? setting.

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:DPATtern:PATTern

Function Sets or queries the data for UART signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:DPATtern:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:DPATtern:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:DPATTERN
:PATTERN "00110101"
:TRIGGER:ATRIGGER:UART:DATA:DPATTERN
:PATTERN?
-> :TRIGGER:ATRIGGER:UART:DATA:DPATT
ERN:PATTERN "00110101"
```

Description The number of bytes that can be set varies depending on the :TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:DBYTe? setting.

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:PFORmat

Function Sets or queries the data input format, which is one of the ID and data conditions, for UART signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:PFORmat {ASCii|PATTern}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:DATA:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:PFORM
AT PATTERN
:TRIGGER:ATRIGGER:UART:DATA:PFORMAT?
-> :TRIGGER:ATRIGGER:UART:DATA:PFORM
AT PATTERN
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor?

Function Queries all UART signal trigger error settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:ERRor?
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor:FRAMing

Function Sets or queries the UART signal trigger framing error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:ERRor:FRAMing {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:ERRor:FRAMing?
```

Example
```
:TRIGGER:ATRIGGER:UART:ERROR:FRAMI
NG ON
:TRIGGER:ATRIGGER:UART:ERROR:FRAMI
NG?
-> :TRIGGER:ATRIGGER:UART:ERROR:FRAM
ING 1
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor:PARity

Function Sets or queries the UART signal trigger parity error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:ERRor:PARity {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:ERRor:PARity?
```

Example
```
:TRIGGER:ATRIGGER:UART:ERROR:PARI
TY ON
:TRIGGER:ATRIGGER:UART:ERROR:PARITY?
-> :TRIGGER:ATRIGGER:UART:ERROR:PARI
TY 1
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor:PMODE

Function Sets or queries the UART signal trigger parity mode setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:ERRor:PMODE {EVEN|ODD}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:ERRor:PMODE?
```

Example
```
:TRIGGER:ATRIGGER:UART:ERROR:PMO
DE EVEN
:TRIGGER:ATRIGGER:UART:ERROR:PMODE?
-> :TRIGGER:ATRIGGER:UART:ERROR:PMO
DE EVEN
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:FORMat

Function Sets or queries the UART signal trigger format setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:FORMat {BIT7Parity|BIT8Parity|BIT8N
oparity}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:FORMat?
```

Example
```
:TRIGGER:ATRIGGER:UART:FORMAT BIT7PA
RITY
:TRIGGER:ATRIGGER:UART:FORMAT?
-> :TRIGGER:ATRIGGER:UART:FORMAT BIT
7PARITY
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:MODE

Function Sets or queries the trigger type for UART signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:MODE {DATA|EDATa|ERRor}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:MODE?
```

Example
```
:TRIGGER:ATRIGGER:UART:MODE DATA
:TRIGGER:ATRIGGER:UART:MODE?
-> :TRIGGER:ATRIGGER:UART:MODE DATA
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:POLarity

Function Sets or queries the UART signal analysis polarity.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:POLarity {NEGative|POSitive}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:UART:POLARITY NEGA
TIVE
:TRIGGER:ATRIGGER:UART:POLARITY?
-> :TRIGGER:ATRIGGER:UART:POLARITY N
EGATIVE
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:SOURce

Function Sets or queries the UART signal analysis signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:SOURce {<NRf>|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:UART:SOURCE 1
:TRIGGER:ATRIGGER:UART:SOURCE?
-> :TRIGGER:ATRIGGER:UART:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:SPOint

Function Sets or queries the UART signal analysis sample point.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART
:SPOint {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:UART
:SPOint?
```
<NRf> = 18.8 to 90.6

Example
```
:TRIGGER:ATRIGGER:UART:SPOINT 18.8
:TRIGGER:ATRIGGER:UART:SPOINT?
-> :TRIGGER:ATRIGGER:UART:SPOINT 18.
8
```

:TRIGger{[:ATRigger]|:BTRigger}:WINDow?

Function Queries all window trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:WIND
ow?
```

:TRIGger{[:ATRigger]|:BTRigger}:WINDow:POLarity

Function Sets or queries the polarity of the window trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:WIND
ow:POLarity {IN|OUT}
:TRIGger{[:ATRigger]|:BTRigger}:WIND
ow:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:WINDOW:POLARITY IN
:TRIGGER:ATRIGGER:WINDOW:POLARITY?
-> :TRIGGER:ATRIGGER:WINDOW:POLARIT
Y IN
```

:TRIGger{[:ATRigger]|:BTRigger}:WINDow:SOURce

Function Sets or queries the window trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:WIND
ow:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:WIND
ow:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:WINDOW:SOURCE 1
:TRIGGER:ATRIGGER:WINDOW:SOURCE?
-> :TRIGGER:ATRIGGER:WINDOW:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:WINDow:TQUalify

Function Sets or queries the time condition for window triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:WINDow:TQUalify {INSide|LTHan|MTHan|NONE|OUTSide|TIMeout}
:TRIGger{[:ATRigger]|:BTRigger}:WINDow:TQUalify?
```

Example
```
:TRIGGER:ATRIGGER:WINDOW:TQUALIFY INSIDE
:TRIGGER:ATRIGGER:WINDOW:TQUALIFY?
-> :TRIGGER:ATRIGGER:WINDOW:TQUALIFY INSIDE
```

:TRIGger{[:ATRigger]|:BTRigger}:WOR?

Function Queries all window OR trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:WOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:WOR:ALL

Function Sets or queries whether to make all channels window OR trigger sources at once.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:WOR:ALL {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:WOR:ALL?
```

Example
```
:TRIGGER:ATRIGGER:WOR:ALL ON
:TRIGGER:ATRIGGER:WOR:ALL?
-> :TRIGGER:ATRIGGER:WOR:ALL 1
```

:TRIGger{[:ATRigger]|:BTRigger}:WOR:CAHNnel<x>

Function Sets or queries the slope of each channel of the window OR trigger.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:WOR:CAHNnel<x> {IN|OFF|OUT}
:TRIGger{[:ATRigger]|:BTRigger}:WOR:CAHNnel<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:WOR:CAHNNEL1 IN
:TRIGGER:ATRIGGER:WOR:CAHNNEL1?
-> :TRIGGER:ATRIGGER:WOR:CAHNNEL1 IN
```

:TRIGger[:ATRigger]:INTerval?

Function Queries all edge interval trigger settings.

Syntax
```
:TRIGger[:ATRigger]:INTerval?
```

:TRIGger[:ATRigger]:INTerval:SLOPe

Function Sets or queries the edge interval trigger slope.

Syntax
```
:TRIGger[:ATRigger]:INTerval:SLOPe {FALL|RISE]
:TRIGger[:ATRigger]:INTerval:SLOPe?
```

Example
```
:TRIGGER:ATRIGGER:INTERVAL:SLOPE FALL
:TRIGGER:ATRIGGER:INTERVAL:SLOPE?
-> :TRIGGER:ATRIGGER:INTERVAL:SLOPE FALL
```

:TRIGger[:ATRigger]:INTerval:SOURce

Function Sets or queries the edge interval trigger source.

Syntax
```
:TRIGger[:ATRigger]:INTerval:SOURce <NRf>|BIT<x>}
:TRIGger[:ATRigger]:INTerval:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:INTERVAL:SOURCE 1
:TRIGGER:ATRIGGER:INTERVAL:SOURCE?
-> :TRIGGER:ATRIGGER:INTERVAL:SOURCE 1
```

:TRIGger[:ATRigger]:INTerval:TQUalify

Function Sets or queries the time condition for edge interval triggering.

Syntax
```
:TRIGger[:ATRigger]:INTerval:TQUalify {INSide|LTHan|MTHan|OUTSide|TIMeout}
:TRIGger[:ATRigger]:INTerval:TQUalify?
```

Example
```
:TRIGGER:ATRIGGER:INTERVAL:TQUALIFY INSIDE
:TRIGGER:ATRIGGER:INTERVAL:TQUALIFY?
-> :TRIGGER:ATRIGGER:INTERVAL:TQUALIFY INSIDE
```

:TRIGger[:ATRigger]:PULSe?

Function Queries all pulse width trigger settings.

Syntax
```
:TRIGger[:ATRigger]:PULSe?
```

:TRIGger[:ATRigger]:PULSe:POLarity

Function Sets or queries the pulse width trigger source polarity.

Syntax
```
:TRIGger[:ATRigger]:PULSe:POLari
ty {NEGative|POSitive}
:TRIGger[:ATRigger]:PULSe:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:PULSE:POLARI
TY NEGATIVE
:TRIGGER:ATRIGGER:PULSE:POLARITY?
-> :TRIGGER:ATRIGGER:PULSE:POLARI
TY NEGATIVE
```

:TRIGger[:ATRigger]:PULSe:SOURce

Function Sets or queries the pulse width trigger source source.

Syntax
```
:TRIGger[:ATRigger]:PULSe:SOUR
ce {<NRf>|BIT<x>}
:TRIGger[:ATRigger]:PULSe:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:PULSE:SOURCE 1
:TRIGGER:ATRIGGER:PULSE:SOURCE?
-> :TRIGGER:ATRIGGER:PULSE:SOURCE 1
```

:TRIGger[:ATRigger]:PULSe:TQUalify

Function Sets or queries the time condition for pulse width triggering.

Syntax
```
:TRIGger[:ATRigger]:PULSe:TQUali
fy {INSide|LTHan|MTHan|OUTSide|TIMeo
ut}
:TRIGger[:ATRigger]:PULSe:TQUalify?
```

Example
```
:TRIGGER:ATRIGGER:PULSE:TQUALI
FY INSIDE
:TRIGGER:ATRIGGER:PULSE:TQUALIFY?
-> :TRIGGER:ATRIGGER:PULSE:TQUALI
FY INSIDE
```

:TRIGger[:ATRigger]:RFTime?

Function Queries all rise/fall time trigger settings.

Syntax
```
:TRIGger[:ATRigger]:RFTime?
```

:TRIGger[:ATRigger]:RFTime:SLOPe

Function Sets or queries the rise/fall time trigger slope.

Syntax
```
:TRIGger[:ATRigger]:RFTime:SLO
Pe {BOTH|FALL|RISE}
:TRIGger[:ATRigger]:RFTime:SLOPe?
```

Example
```
:TRIGGER:ATRIGGER:RFTIME:SLOPE FALL
:TRIGGER:ATRIGGER:RFTIME:SLOPE?
-> :TRIGGER:ATRIGGER:RFTIME:SLOPE FA
LL
```

:TRIGger[:ATRigger]:RFTime:SOURce

Function Sets or queries the rise/fall time trigger source.

Syntax
```
:TRIGger[:ATRigger]:RFTime:SOURce {<
NRf>}
:TRIGger[:ATRigger]:RFTime:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:RFTIME:SOURCE 1
:TRIGGER:ATRIGGER:RFTIME:SOURCE?
-> :TRIGGER:ATRIGGER:RFTIME:SOURCE 1
```

:TRIGger[:ATRigger]:RFTime:TQUalify

Function Sets or queries the time condition for rise/fall time triggering.

Syntax
```
:TRIGger[:ATRigger]:RFTime:TQUali
fy {INSide|LTHan|MTHan|OUTSide}
:TRIGger[:ATRigger]:RFTime:TQUalify?
```

Example
```
:TRIGGER:ATRIGGER:RFTIME:TQUALIFY IN
SIDE
:TRIGGER:ATRIGGER:RFTIME:TQUALIFY?
-> :TRIGGER:ATRIGGER:RFTIME:TQUALIF
Y INSIDE
```

:TRIGger[:ATRigger]:RUNT?

Function Queries all runt trigger settings.

Syntax
```
:TRIGger[:ATRigger]:RUNT?
```

:TRIGger[:ATRigger]:RUNT:POLarity

Function Sets or queries the polarity of the runt trigger source.

Syntax
```
:TRIGger[:ATRigger]:RUNT:POLarity {E
ITHer|NEGative|POSitive}
:TRIGger[:ATRigger]:RUNT:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:RUNT:POLARITY NEGA
TIVE
:TRIGGER:ATRIGGER:RUNT:POLARITY?
-> :TRIGGER:ATRIGGER:RUNT:POLARITY N
EGATIVE
```

:TRIGger[:ATRigger]:RUNT:SOURce

Function Sets or queries the runt trigger source.

Syntax
```
:TRIGger[:ATRigger]:RUNT:SOURce {<NR
f>]
:TRIGger[:ATRigger]:RUNT:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:RUNT:SOURCE 1
:TRIGGER:ATRIGGER:RUNT:SOURCE?
-> :TRIGGER:ATRIGGER:RUNT:SOURCE 1
```

:TRIGger[:ATRigger]:RUNT:TQUalify

Function Sets or queries the time condition for runt triggering.

Syntax
```
:TRIGger[:ATRigger]:RUNT:
TQUalify {INSide|LTHan|MTHan|NONE|OU
TSide}
:TRIGger[:ATRigger]:RUNT:TQUalify?
```

Example
```
:TRIGGER:ATRIGGER:RUNT:TQUALIFY INSI
DE
:TRIGGER:ATRIGGER:RUNT:TQUALIFY?
-> :TRIGGER:ATRIGGER:RUNT:TQUALIFY I
NSIDE
```

:TRIGger[:ATRigger]:TIMeout?

Function Queries all timeout period trigger settings.

Syntax
```
:TRIGger[:ATRigger]:TIMeout?
```

:TRIGger[:ATRigger]:TIMeout:POLarity

Function Sets or queries the polarity of the timeout period trigger source.

Syntax
```
:TRIGger[:ATRigger]:TIMeout:POLari
ty {EITHer|NEGative|POSitive}
:TRIGger[:ATRigger]:TIMeout:POLari
ty?
```

Example
```
:TRIGGER:ATRIGGER:TIMEOUT:POLARI
TY NEGATIVE
:TRIGGER:ATRIGGER:TIMEOUT:POLARITY?
-> :TRIGGER:ATRIGGER:TIMEOUT:POLARI
TY NEGATIVE
```

:TRIGger[:ATRigger]:TIMeout:SOURce

Function Sets or queries the timeout period trigger source.

Syntax
```
:TRIGger[:ATRigger]:TIMeout:SOUR
ce {<NRf>|BIT<x>}
:TRIGger[:ATRigger]:TIMeout:SOURce?
```
<NRf> = 1 to 4
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:TIMEOUT:SOURCE 1
:TRIGGER:ATRIGGER:TIMEOUT:SOURCE?
-> :TRIGGER:ATRIGGER:TIMEOUT:SOUR
CE 1
```

:TRIGger[:ATRigger]:TV?

Function Queries all TV trigger settings.

Syntax
```
:TRIGger[:ATRigger]:TV?
```

:TRIGger[:ATRigger]:TV:FIELd

Function Sets or queries the field that the TV trigger is to be activated on.

Syntax
```
:TRIGger[:ATRigger]:TV:FIELd {DONTca
re|<NRf>}
:TRIGger[:ATRigger]:TV:FIELd?
```
<NRf> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:TV:FIELD 1
:TRIGGER:ATRIGGER:TV:FIELD?
-> :TRIGGER:ATRIGGER:TV:FIELD 1
```

:TRIGger[:ATRigger]:TV:FRAMe

Function Sets or queries the TV trigger frame skip function.

Syntax
```
:TRIGger[:ATRigger]:TV:FRAMe {<NRf>}
:TRIGger[:ATRigger]:TV:FRAMe?
```
<NRf> = 1 or 2, 4, 8

Example
```
:TRIGGER:ATRIGGER:TV:FRAME 2
:TRIGGER:ATRIGGER:TV:FRAME?
-> :TRIGGER:ATRIGGER:TV:FRAME 2
```

:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PAL|SDTV|USERdefine}?

Function Queries all TV trigger mode settings.

Syntax
```
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PA
L|SDTV|USERdefine}?
```

:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PAL|SDTV|USERdefine}:LINE

Function Sets or queries the line that the TV trigger is to be activated on.

Syntax
```
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PA
L|SDTV|USERdefine}:LINE {<NRf>}
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PA
L|SDTV|USERdefine}:LINE?
```
<NRf> = 2 to 2251

Example The following example is for HDTV.
```
:TRIGGER:ATRIGGER:TV:HDTV:LINE 2
:TRIGGER:ATRIGGER:TV:HDTV:LINE?
-> :TRIGGER:ATRIGGER:TV:HDTV:LINE 2
```

:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PAL|SDTV|USERdefine}:POLarity

Function Sets or queries the TV trigger input polarity.

Syntax
```
:TRIGger[:ATRigger]:TV:HDTV:POLarity {POSitive|NEGative}
:TRIGger[:ATRigger]:TV:HDTV:POLarity?
```

Example The following example is for HDTV.
```
:TRIGGER:ATRIGGER:TV:HDTV:POLARITY POSITIVE
:TRIGGER:ATRIGGER:TV:HDTV:POLARITY?
-> :TRIGGER:ATRIGGER:TV:HDTV:POLARITY POSITIVE
```

:TRIGger[:ATRigger]:TV:LEVel

Function Sets or queries the TV trigger level.

Syntax
```
:TRIGger[:ATRigger]:TV:LEVel {<NRf>}
:TRIGger[:ATRigger]:TV:LEVel?
```
<NRf> = 0.1 or 2.0div (in 0.1 div steps)

Example
```
:TRIGGER:ATRIGGER:TV:LEVEL 0.1
:TRIGGER:ATRIGGER:TV:LEVEL?
-> :TRIGGER:ATRIGGER:TV:LEVEL 0.1
```

:TRIGger[:ATRigger]:TV:LFORmat

Function Sets or queries the line designation method for TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:LFORmat {ALL|ONE}
:TRIGger[:ATRigger]:TV:LFORmat?
```

Example
```
:TRIGGER:ATRIGGER:TV:LFORMAT ALL
:TRIGGER:ATRIGGER:TV:LFORMAT?
-> :TRIGGER:ATRIGGER:TV:LFORMAT ALL
```

:TRIGger[:ATRigger]:TV:SOURce

Function Sets or queries the TV trigger source.

Syntax
```
:TRIGger[:ATRigger]:TV:SOURce {<NRf>}
:TRIGger[:ATRigger]:TV:SOURce?
```
<NRf> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:TV:SOURCE 1
:TRIGGER:ATRIGGER:TV:SOURCE?
-> :TRIGGER:ATRIGGER:TV:SOURCE 1
```

:TRIGger[:ATRigger]:TV:TYPE

Function Sets or queries the TV trigger input type.

Syntax
```
:TRIGger[:ATRigger]:TV:TYPE {NTSC|HDTV[,{I1080_50|I1080_60|P1080_24|P1080_25|P1080_60|P720_60|SF1080_24}]|PAL|SDTV|USERdefine}
:TRIGger[:ATRigger]:TV:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:TV:TYPE NTSC
:TRIGGER:ATRIGGER:TV:TYPE?
-> :TRIGGER:ATRIGGER:TV:TYPE NTSC
```

:TRIGger[:ATRigger]:TV:USERdefine?

Function Queries all user-defined TV trigger settings.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine?
```

:TRIGger[:ATRigger]:TV:USERdefine:DEFinition

Function Sets or queries the user-defined TV trigger definition.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:DEFinition {HD|SD}
:TRIGger[:ATRigger]:TV:USERdefine:DEFinition?
```

Example
```
:TRIGGER:ATRIGGER:TV:USERDEFINE:DEFINITION HD
:TRIGGER:ATRIGGER:TV:USERDEFINE:DEFINITION?
-> :TRIGGER:ATRIGGER:TV:USERDEFINE:DEFINITION HD
```

:TRIGger[:ATRigger]:TV:USERdefine:HFRejection (HighFrequencyREJECTION)

Function Sets or queries the on/off state of the low-pass filter (HF rejection) for user-defined TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:HFRejection {<Frequency>|OFF}
:TRIGger[:ATRigger]:TV:USERdefine:HFRejection?
```
<Frequency> = 300kHz

Example
```
:TRIGGER:ATRIGGER:TV:USERDEFINE:HFREJECTION 300KHZ
:TRIGGER:ATRIGGER:TV:USERDEFINE:HFREJECTION?
-> :TRIGGER:ATRIGGER:TV:USERDEFINE:HFREJECTION 300000
```

:TRIGger[:ATRigger]:TV:USERdefine:HSYNc (Hsync Freq)

Function Sets or queries the horizontal sync signal frequency for user-defined TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:HSYNc {<Frequency>}
:TRIGger[:ATRigger]:TV:USERdefine:HSYNc?
```
<Frequency> = 10k to 200k (Hz)

Example
```
:TRIGGER:ATRIGGER:TV:USERDEFINE:HSYNC 10KHZ
:TRIGGER:ATRIGGER:TV:USERDEFINE:HSYNC?
-> :TRIGGER:ATRIGGER:TV:USERDEFINE:HSYNC 10.000E+03
```

:TRIGger[:ATRigger]:TV:USERdefine:LINE

Function Sets or queries the line number for user-defined TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:LINE {<NRf>}
:TRIGger[:ATRigger]:TV:USERdefine:LINE?
```
<NRf> = 2 to 2251

Example
```
:TRIGGER:ATRIGGER:TV:USERDEFINE:LINE 2
:TRIGGER:ATRIGGER:TV:USERDEFINE:LINE?
-> :TRIGGER:ATRIGGER:TV:USERDEFINE:LINE 2
```

:TRIGger[:ATRigger]:TV:USERdefine:POLarity

Function Sets or queries the input polarity for user-defined TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:POLarity {POSitive|NEGative}
:TRIGger[:ATRigger]:TV:USERdefine:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:TV:USERDEFINE:POLARITY POSITIVE
:TRIGGER:ATRIGGER:TV:USERDEFINE:POLARITY?
-> :TRIGGER:ATRIGGER:TV:USERDEFINE:POLARITY POSITIVE
```

:TRIGger[:ATRigger]:TV:USERdefine:SGUard

Function Sets or queries the sync guard for user-defined TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:SGUard {<NRf>}
:TRIGger[:ATRigger]:TV:USERdefine:SGUard?
```
<NRf> = 60 to 90 (%)

Example
```
:TRIGGER:ATRIGGER:TV:USERDEFINE:SGUARD 60
:TRIGGER:ATRIGGER:TV:USERDEFINE:SGUARD?
-> :TRIGGER:ATRIGGER:TV:USERDEFINE:SGUARD 60
```

:TRIGger:COMBination

Function Sets or queries the trigger combination.

Syntax
```
:TRIGger:COMBination {ABN|ADB|OFF}
:TRIGger:COMBination?
```

Example
```
:TRIGGER:COMBINATION ABN
:TRIGGER:COMBINATION?
-> :TRIGGER:COMBINATION ABN
```

:TRIGger:DELay?

Function Queries all delay settings.

Syntax
```
:TRIGger:DELay?
```

:TRIGger:DELay:TIME

Function Sets or queries the delay (time from the trigger point to the trigger position).

Syntax
```
:TRIGger:DELay:TIME {<Time>}
:TRIGger:DELay:TIME?
```
<Time> = 0 to 10s (in 1/sample rate steps)

Example
```
:TRIGGER:DELAY:TIME 2US
:TRIGGER:DELAY:TIME?
-> :TRIGGER:DELAY:TIME 2.00E-06
```

Description The value is fixed to 0 when the time base is set to external clock. On this instrument, set the trigger delay using the time from the trigger point to the trigger position. Use :TRIGger:POSition to set or query the trigger position.

:TRIGger:FORCe

Function Forces the instrument to trigger.

Syntax
```
:TRIGger:FORCe
```

Example
```
:TRIGGER:FORCE
```

:TRIGger:HOLDoff?
Function Queries all hold-off settings.
Syntax :TRIGger:HOLDoff?

:TRIGger:HOLDoff:TIME
Function Sets or queries the hold-off time.
Syntax :TRIGger:HOLDoff:TIME {<Time>}
:TRIGger:HOLDoff:TIME?
<Time> = 20 ns to 10 s
Example :TRIGGER:HOLDOFF:TIME 20NS
:TRIGGER:HOLDOFF:TIME?
-> :TRIGGER:HOLDOFF:TIME 20.0E-09

:TRIGger:MODE
Function Sets or queries the trigger mode.
Syntax :TRIGger:MODE {AUTO|ALEVel|NORMal|NSINgle}
:TRIGger:MODE?
Example :TRIGGER:MODE AUTO
:TRIGGER:MODE?
-> :TRIGGER:MODE AUTO
Description Use :TRIGger:SCOunt to set the count for NSINgle.

:TRIGger:POSition
Function Sets or queries trigger position.
Syntax :TRIGger:POSition {<NRf>}
:TRIGger:POSition?
<NRf> = 0 to 100 (%, in steps of 0.1)
Example :TRIGGER:POSITION 50
:TRIGGER:POSITION?
-> :TRIGGER:POSITION 50.000

:TRIGger:SCOunt
Function Sets or queries the number of times the trigger condition is to be met when the trigger mode is set to N Single.
Syntax :TRIGger:SCOunt {<NRf>}
:TRIGger:SCOunt?
<NRf> = 1 or 20000
Example :TRIGGER:SCOUNT 100
:TRIGGER:SCOUNT?
-> :TRIGGER:SCOUNT 100

:TRIGger:SOURce?
Function Queries all enhanced trigger source settings.
Syntax :TRIGger:SOURce?

:TRIGger:SOURce:CHANnel<x>?
Function Queries all settings related to the specified trigger source for enhanced triggering.
Syntax :TRIGger:SOURce:CHANnel<x>?
<x> = 1 to 4

:TRIGger:SOURce:CHANnel<x>:HFRejection (HighFrequencyREJECTION)
Function Sets or queries the low-pass filter of the specified trigger source for enhanced triggering.
Syntax :TRIGger:SOURce:CHANnel<x>:HFRejection {<Frequency>|OFF}
:TRIGger:SOURce:CHANnel<x>:HFRejection?
<x> = 1 to 4
<Frequency> = 15KHz to 20MHz
Example :TRIGGER:SOURCE:CHANNEL1:HFREJECTION 15KHZ
:TRIGGER:SOURCE:CHANNEL1:HFREJECTION?
-> :TRIGGER:SOURCE:CHANNEL1:HFREJECTION 15000

:TRIGger:SOURce:CHANnel<x>:NREJection
Function Sets or queries the noise rejection of the specified trigger source for enhanced triggering.
Syntax :TRIGger:SOURce:CHANnel<x>:NREJection {HIGH|LOW|MIDDle}
:TRIGger:SOURce:CHANnel<x>:NREJection?
<x> = 1 to 4
Example :TRIGGER:SOURCE:CHANNEL1:NREJECTION HIGH
:TRIGGER:SOURCE:CHANNEL1:NREJECTION?
-> :TRIGGER:SOURCE:CHANNEL1:NREJECTION HIGH

:TRIGger:SOURce:CHANnel<x>:LEVel
Function Sets or queries the trigger level of the specified trigger source for enhanced triggering.
Syntax :TRIGger:SOURce:CHANnel<x>:LEVel {<Voltage>}
:TRIGger:SOURce:CHANnel<x>:LEVel?
<x> = 1 to 4
<Voltage> = See the Features Guide for this information.
Example :TRIGGER:SOURCE:CHANNEL1:LEVEL 0V
:TRIGGER:SOURCE:CHANNEL1:LEVEL?
-> :TRIGGER:SOURCE:CHANNEL1:LEVEL 0.000E+00
Description When a current probe is in use, this command does not set or query the <Current> value.

:TRIGger:SOURce:CHANnel<x>:ULLeVel

Function Sets or queries the upper and lower limits of the enhanced trigger source level.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:ULLeV
el {<Voltage>,<Voltage>}
:TRIGger:SOURce:CHANnel<x>:ULLeVel?
```
<x> = 1 to 4
<Voltage> = See the Features Guide for this information.

Example
```
:TRIGGER:SOURCE:CHANNEL1:ULLEV
EL 0.5V,-0.5V
:TRIGGER:SOURCE:CHANNEL1:ULLEVEL?
-> :TRIGGER:SOURCE:CHANNEL1:ULLEV
EL 0.500E+00,-0.500E+00
```

:TRIGger:TIME<x>

Function Sets or queries the time for enhanced triggering.

Syntax
```
:TRIGger:TIME<x> {<Time>}
:TRIGger:TIME<x>?
```
<x> = 1 or 2
<Time> = 4ns to 10s (in 2ns steps)

Example
```
:TRIGGER:TIME1 1S
:TRIGGER:TIME1?
-> :TRIGGER:TIME1 1S
```

5.34 WAVeform Group

The commands in this group deal with the acquired waveform data. There are no front panel keys that correspond to the commands in this group.

:WAVeform?

Function Queries all the waveform data information.

Syntax `:WAVeform?`

:WAVeform:ALL?

Function Queries all settings related to the :WAVeform:ALL:SEND? query.

Syntax `:WAVeform:ALL?`

:WAVeform:ALL:SEND?

Function Queries the waveform data specified by :WAVeform:ALL:TRACe.

Syntax `:WAVeform:ALL:SEND? {<NRf>}`
<NRf>= 0 to the acquisition count

Example
```
:WAVEFORM:ALL:SEND? 0
-> #800007558
03 00 85 00 00 00 00 00 00 00 00 00
00 00 E2 04
00 00 00 00 00 00 E2 04 00 00 62 00
AC 11 44 00
6C 00 26 00 08 00 3C 18 3E 18 4E 18
0A 18 18 18 ...
```

Description
- You can query waveform data during measurement.
- You cannot query waveform data when the trigger mode is set to Single or NSingle.
- You cannot query waveform data when the display is in roll mode.
- You cannot query waveform data when continuous statistical processing of automatically measured waveform parameters or user-defined computation is enabled.
- The data format is shown below.
  #8 (eight digits for the number of transmission data bytes in decimal format) <- Block data header
  (two bytes for the number of traces)(eight bytes for the acquisition count) <- Common information
  (four bytes for the trace number)(eight bytes for the reserved area)(four bytes for the number of data points) <- Trace information 1
  (data sequence) . . .
  (four bytes for the trace number)(eight bytes for the reserved area)(four bytes for the number of data points) <- Trace information 2
  (data sequence) . . .
  . . . (repeated for all the traces)
- The common information and trace information is written in little-endian byte order.
- The data sequence format is WORD. The byte order is little endian.
- You can query up to 12.5 kPoints in a record.
- <NRf> You can acquire the latest waveform data by setting (the acquisition count) to 0.
- To acquire waveforms so that the same waveform data is not acquired, specify the acquisition count of the common information acquired with this command the last time using <NRf>. If the specified acquisition count and the DLM200's acquisition count are the same at the point in time that the query is received, #800000000 is returned.

:WAVeform:ALL:TRACe

Function Sets or queries the target waveform for the :WAVeform:ALL:SEND? command.

Syntax `:WAVeform:ALL:TRACe {<NRf>|ALL|LOGic|MATH<x>}:WAVeform:ALL:TRACe?`
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:WAVEFORM:ALL:TRACE 1
:WAVEFORM:ALL:TRACE?
-> :WAVEFORM:ALL:TRACE 1
```

:WAVeform:BITS?

Function Queries the bit length of the waveform data specified by :WAVeform:TRACe.

Syntax `:WAVeform:BITS?`

Example
```
:WAVEFORM:BITS?
-> :WAVEFORM:BITS 16
```

:WAVeform:BYTeorder

Function Sets or queries the transmission byte order for waveform data in word format, which is 2 bytes or longer in length.

Syntax `:WAVeform:BYTeorder {LSBFirst|MSBFirst}`
`:WAVeform:BYTeorder?`

Example
```
:WAVEFORM:BYTEORDER LSBFIRST
:WAVEFORM:BYTEORDER?
-> :WAVEFORM:BYTEORDER LSBFIRST
```

:WAVeform:END

Function Sets or queries the end data point in the waveform specified by :WAVeform:TRACe.

Syntax `:WAVeform:END {<NRf>}`
`:WAVeform:END?`
<NRf> = 0 to 999999999

Example
```
:WAVEFORM:END 12499
:WAVEFORM:END?
-> :WAVEFORM:END 12499
```

Description Use :WAVeform:LENGth? to query the total number of data points.

:WAVeform:FORMat

Function Sets or queries the format of the waveform data to be sent.

Syntax `:WAVeform:FORMat {ASCii|BYTE|RBYTe|WORD}`
`:WAVeform:FORMat?`

Example
```
:WAVEFORM:FORMAT ASCII
:WAVEFORM:FORMAT?
-> :WAVEFORM:FORMAT ASCII
```

Description For information about the differences depending on this format, see the description of the :WAVeform:SEND? command.

:WAVeform:LENGth?

Function Queries the total number of points of the waveform specified by :WAVeform:TRACe.

Syntax `:WAVeform:LENGth?`

Example
```
:WAVEFORM:LENGTH?
-> :WAVEFORM:LENGTH 12500
```

:WAVeform:OFFSet?

Function Queries the offset value used to convert the waveform data specified by :WAVeform:TRACe to physical values.

Syntax `:WAVeform:OFFSet?`

Example
```
:WAVEFORM:OFFSET?
-> :WAVEFORM:OFFSET 0.000E+00
```

Description
- This offset is used when converting the <block data> output by the :WAVeform:SEND? command into physical values.
- When :SYSTem:OCANcel is set to ON, the instrument returns 0.

:WAVeform:POSition?

Function Queries the vertical position used to convert waveform data into voltage when :WAVeform:FORMat is set to RBYTe.

Syntax `:WAVeform:POSition?`

Example
```
:WAVEFORM:POSITION?
-> :WAVEFORM:POSITION 128
```

:WAVeform:RANGe?

Function Queries the range value used to convert the waveform data specified by :WAVeform:TRACe to physical values.

Syntax `:WAVeform:RANGe?`

Example
```
:WAVEFORM:RANGE?
-> :WAVEFORM:RANGE 5.000E+00
```

Description This range is used when converting the <block data> output by the :WAVeform:SEND? command into physical values.

:WAVeform:RECord

Function Sets or queries the record number that WAVeform commands will be applied to.

Syntax `:WAVeform:RECord {<NRf>|AVERage|MINimum}`
`:WAVeform:RECord?`
<NRf> = 0 to –99999

Example
```
:WAVEFORM:RECORD 0
:WAVEFORM:RECORD?
-> :WAVEFORM:RECORD 0
```

Description If AVERage is specified, the command will be applied to history average waveform. Specify “MINimum” to specify the minimum record number. The settable record numbers vary depending on the acquisition settings. For details, see the Features Guide.

:WAVeform:RECord? MINimum

Function Queries the smallest record number in the source channel history.

Syntax `:WAVeform:RECord? MINimum`

Example `:WAVEFORM:RECORD? MINIMUM`
`-> :WAVEFORM:RECORD -49999`

:WAVeform:SEND?

Function Queries the waveform data specified by :WAVeform:TRACe.

Syntax `:WAVeform:SEND? [<NRf>]`
<NRf> = 1 to 100000
The <NRf> range varies depending on the record length setting.

Example `:WAVEFORM:SEND?`
`-> #8(8-digit number of bytes)(data sequence) or <NRf>,<NRf>,...`

Description
- The :WAVeform:SEND? output format varies depending on the :WAVeform:FORMat setting.
  (1) When set to “ASCii,” the information is returned in this form: <Voltage>,<Voltage>,...<Voltage>
  (2) When set to “BYTE” or “Word,” the information is returned in <block data> format. The following equation can be used to make the conversion. Voltage (computed value) = (range×data÷Division*) + offset
  * “BYTE”: Division = 12.5
    “WORD”: Division = 3200
  (3) When set to “RBYTe,” the information is returned in <block data> format. The following equation can be used to make the conversion. Voltage (computed value) = (range×(data – Position) ÷Division*) + offset
  * Division = 25
    Position = The return value of “: WAVeform: POSition?”
- <NRf> can be omitted. If you specify <NRf>, waveform data is queried <NRf> times in order starting from the record located at the record number specified by :WAVeform:RECord – <NRf>.

:WAVeform:SIGN?

Function Queries whether signs are included in the binary data of the source waveform specified by :WAVeform:TRACe when the data is queried.

Syntax `:WAVeform:SIGN?`

Example `:WAVEFORM:SIGN?`
`-> :WAVEFORM:SIGN 1`

:WAVeform:SRATe? (Sample RATE)

Function Queries the sample rate of the record specified by :WAVeform:RECord.

Syntax `:WAVeform:SRATe?`

Example `:WAVEFORM:SRATE?`
`-> :WAVEFORM:SRATE 1.25E+09`

:WAVeform:STARt

Function Sets or queries the start data point in the waveform specified by :WAVeform:TRACe.

Syntax `:WAVeform:STARt {<NRf>}`
`:WAVeform:STARt?`
`<NRf> = 0 to 499999999`

Example `:WAVEFORM:START 0`
`:WAVEFORM:START?`
`-> :WAVEFORM:START 0`

:WAVeform:TRACe

Function Sets or queries the waveform that WAVeform commands will be applied to.

Syntax `:WAVeform:TRACe {<NRf>|LOGic|MATH<x>}`
`:WAVeform:TRACe?`
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<x> = 1 to 4 (1 or 2 on 2-channel models)

Example `:WAVEFORM:TRACE 1`
`:WAVEFORM:TRACE?`
`-> :WAVEFORM:TRACE 1`

:WAVeform:TRIGger?

Function Queries the trigger position of the record specified by :WAVeform:RECord.

Syntax `:WAVeform:TRIGger?`

Example `:WAVEFORM:TRIGGER?`
`-> :WAVEFORM:TRIGGER 6250`

Description Queries the number of points from the beginning of the record length to the trigger position.

:WAVeform:TYPE?

Function Queries the acquisition mode of the waveform specified by :WAVeform:TRACe.

Syntax `:WAVeform:TYPE?`

Example `:WAVEFORM:TYPE?`
`-> :WAVEFORM:TYPE NORMAL`

5.35 WPARameter Group

:WPARameter<x>?

Function Queries all waveform parameter measurement settings.

Syntax `:WPARameter<x>?`
<x> = 1 or 2 (1 only for 2-channel models)

:WPARameter<x>:DISPlay

Function Sets or queries the on/off state of waveform parameter measurement display.

Syntax `:WPARameter<x>:DISPlay {<Boolean>}`
`:WPARameter<x>:DISPlay?`
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:DISPLAY ON
:WPARAMETER1:DISPLAY?
-> :WPARAMETER1:DISPLAY 1
```

:WPARameter<x>:HISTogram?

Function Queries all settings related to the histogram display of waveform parameter measurement.

Syntax `:WPARameter<x>:HISTogram?`
<x> = 1 or 2 (1 only for 2-channel models)

:WPARameter<x>:HISTogram:MEASure?

Function Queries all settings related to the automated measurement on the histogram display of waveform parameter measurement.

Syntax `:WPARameter<x>:HISTogram:MEASure?`
<x> = 1 or 2 (1 only for 2-channel models)

:WPARameter<x>:HISTogram:MEASure:MODE

Function Sets or queries the auto measurement mode of the histogram display of waveform parameter measurement.

Syntax `:WPARameter<x>:HISTogram:MEASure:MODE {OFF|PARameter}`
`:WPARameter<x>:HISTogram:MEASure:MODE?`
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:HISTOGRAM:MEASURE:MODE OFF
:WPARAMETER1:HISTOGRAM:MEASURE:MODE?
-> :WPARAMETER1:HISTOGRAM:MEASURE:MODE OFF
```

:WPARameter<x>:HISTogram:MEASure:PARameter?

Function Queries all settings related to the automated measurement of the histogram parameters of waveform parameter measurement.

Syntax `:WPARameter<x>:HISTogram:MEASure:PARameter?`
<x> = 1 or 2 (1 only for 2-channel models)

:WPARameter<x>:HISTogram:MEASure:PARameter:ALL

Function Collectively turns on or off all the histogram parameters of waveform parameter measurement.

Syntax `:WPARameter<x>:HISTogram:MEASure:PARameter:ALL {<Boolean>}`
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER:ALL ON
```

:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>?

Function Queries all settings related to the histogram parameters of waveform parameter measurement.

Syntax `:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>?`
<x> = 1 or 2 (1 only for 2-channel models)
<Parameter> = {C1|C2|DC|MAXimum|MEAN|MEDian|MINimum|PEAK|SD2integ|SD3integ|SDEViation|SDINteg}

:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>:STATe

Function Sets or queries the on/off state the histogram parameters of waveform parameter measurement display.

Syntax `:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>:STATe {<Boolean>}`
`:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>:STATe?`
<x> = 1 or 2 (1 only for 2-channel models)
<Parameter> = {C1|C2|DC|MAXimum|MEAN|MEDian|MINimum|PEAK|SD2integ|SD3integ|SDEViation|SDINteg}

Example The following example is for the maximum value.
```
:WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER1:MAXIMUM:STATE ON
:WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER1:MAXIMUM:STATE?
-> :WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER1:MAXIMUM:STATE 1
```

:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>:VALue?

Function Queries the measured value of a histogram parameter of waveform parameter measurement.

Syntax
```
:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Parameter> = {C1|C2|DC|MAXimum|MEAN|MEDian|MINimum|PEAK|SD2integ|SD3integ|SDEViation|SDINteg}

Example The following example is for the maximum value.
```
:WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER:MAXIMUM:VALUE?
-> :WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER:MAXIMUM:VALUE 1.000E+00
```

:WPARameter<x>:HISTogram:MEASure:PARameter:POSition<y>

Function Sets or queries the position a parameter of waveform parameter measurement histogram.

Syntax
```
:WPARameter<x>:HISTogram:MEASure:PARameter:POSition<y> {<NRf>}
:WPARameter<x>:HISTogram:MEASure:PARameter:POSition<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = –5 div to 5 div

Example
```
:WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER1:POSITION 1
:WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER1:POSITION?
-> :WPARAMETER1:HISTOGRAM:MEASURE:PARAMETER1:POSITION 1.000E+00
```

:WPARameter<x>:ITEM

Function Sets or queries a waveform parameter of waveform parameter measurement.

Syntax
```
:WPARameter<x>:ITEM {<NRf>|BIT<y>|CALC<y>|MATH<y>,<Parameter>[,2]}
:WPARameter<x>:ITEM?
```
<x> of WPARameter<x> = 1 or 2 (1 on 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of BIT<y> = 1 to 8
<y> of CALC<y> = 1 to 4
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)
<Parameter> = {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|I2T|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|P|PABS|PERiod|PN|PNUMber|PON|POVershoot|PP|PTOFf|PTON|PTOPeak|PTOTal|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2|WH|WHABs|WHN|WHON|WHP|WHTOFf|WHTON|WHTOTal|Z}

Example
```
:WPARAMETER1:ITEM 1,AVERAGE
:WPARAMETER1:ITEM?
-> :WPARAMETER1:ITEM 1,AVERAGE
```

Description
- For BIT<y>, the available <Parameter> values are {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}.
- For CALC<y>, the parameter is not necessary.l
- For <NRf> and MATH<y>, the available <Parameter> values are {AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DT|DUTYcycle|ENUMber|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2}.
- When <Parameter > is set to {PON|PTOFf|PTON|PTOTal|WHON|WHTOFf|WHTON|WHTOTal}, the [,2] after the <Parameter> cannot be specified.

:WPARameter<x>:MODE

Function Sets or queries the waveform parameter measurement mode.

Syntax
```
:WPARameter<x>:MODE {HISTogram|TRENd}
:WPARameter<x>:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:MODE HISTOGRAM
:WPARAMETER1:MODE?
-> :WPARAMETER1:MODE HISTOGRAM
```

:WPARameter<x>:TRENd?

Function Queries all settings related to the trend display of waveform parameter measurement.

Syntax
```
:WPARameter<x>:TRENd?
```
<x> = 1 or 2 (1 only for 2-channel models)

:WPARameter<x>:TRENd:ASCale

Function Executes auto scaling of the trend display of waveform parameter measurement.

Syntax `:WPARameter<x>:TRENd:ASCale`
<x> = 1 or 2 (1 only for 2-channel models)

Example `:WPARAMETER1:TREND:ASCALE`

:WPARameter<x>:TRENd:CURSor?

Function Queries all settings related to the cursor measurement on the trend display of waveform parameter measurement.

Syntax `:WPARameter<x>:TRENd:CURSor?`
<x> = 1 or 2 (1 only for 2-channel models)

:WPARameter<x>:TRENd:CURSor:C<y>?

Function Queries the measured value of a cursor of waveform parameter measurement.

Syntax `:WPARameter<x>:TRENd:CURSor:C<y>?`
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:WPARAMETER1:TREND:CURSOR:C1?
-> :WPARAMETER1:TREND:CURSOR:C1 1.000E+00
```

:WPARameter<x>:TRENd:CURSor:DC?

Function Queries the measured value of a cursor of the trend of waveform parameter measurement.

Syntax `:WPARameter<x>:TRENd:CURSor:DC?`
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:TREND:CURSOR:DC?
-> :WPARAMETER1:TREND:CURSOR:DC 1.000E+00
```

:WPARameter<x>:TRENd:CURSor:MODE

Function Sets or queries the auto measurement mode of the trend of waveform parameter measurement.

Syntax
```
:WPARameter<x>:TRENd:CURSor:MODE {<Boolean>}
:WPARameter<x>:TRENd:CURSor:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:TREND:CURSOR:MODE ON
:WPARAMETER1:TREND:CURSOR:MODE?
-> :WPARAMETER1:TREND:CURSOR:MODE 1
```

:WPARameter<x>:TRENd:CURSor:POSition<y>

Function Sets or queries the position a cursor of the trend of waveform parameter measurement display.

Syntax
```
:WPARameter<x>:TRENd:CURSor:POSition<y> {<NRf>}
:WPARameter<x>:TRENd:CURSor:POSition<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = –5 div to 5 div

Example
```
:WPARAMETER1:TREND:CURSOR:POSITION1 1
:WPARAMETER1:TREND:CURSOR:POSITION1?
-> :WPARAMETER1:TREND:CURSOR:POSITION1 1.000E+00
```

:WPARameter<x>:TRENd:HRANge

Function Sets or queries the trend display source window for waveform parameter measurement.

Syntax
```
:WPARameter<x>:TRENd:HRANge {MAIN|Z1|Z2}
:WPARameter<x>:TRENd:HRANge?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:TREND:HRANGE MAIN
:WPARAMETER1:TREND:HRANGE?
-> :WPARAMETER1:TREND:HRANGE MAIN
```

:WPARameter<x>:TRENd:HSPan

Function Sets or queries the trend display horizontal span for waveform parameter measurement.

Syntax
```
:WPARameter<x>:TRENd:HSPan {<NRf>}
:WPARameter<x>:TRENd:HSPan?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 100000

Example
```
:WPARAMETER1:TREND:HSPAN 1
:WPARAMETER1:TREND:HSPAN?
-> :WPARAMETER1:TREND:HSPAN 1
```

:WPARameter<x>:TRENd:VERTical

Function Sets or queries the vertical range of the trend of waveform parameter measurement.

Syntax
```
:WPARameter<x>:TRENd:VERTical {<NRf>,<NRf>}
:WPARameter<x>:TRENd:VERTical?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:WPARAMETER1:TREND:VERTICAL 1
:WPARAMETER1:TREND:VERTICAL?
-> :WPARAMETER1:TREND:VERTICAL 1.00000E+00
```

:WPARameter<x>:VTDisplay

Function Sets or queries the VT waveform display on/off state.

Syntax
```
:WPARameter<x>:VTDisplay {<Boolean>}
:WPARameter<x>:VTDisplay?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:WPARAMETER1:VTDISPLAY ON
:WPARAMETER1:VTDISPLAY?
-> :WPARAMETER1:VTDISPLAY 1
```

:WPARameter<x>:WAIT?

Function Waits for the completion of automated measurement with a timeout.

Syntax
```
:WPARameter<x>:WAIT? {<NRf>}
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 36000 (timeout period, 100 ms)

Example
```
:WPARAMETER1:WAIT?
-> :WPARAMETER1:WAIT 1
```

Description If the execution of automated measurement is completed within the timeout period, 0 is returned. If it is not completed or if automated measurement is not being performed, 1 is returned. Even if you set a long timeout period, 0 is returned as soon as the automated measurement is completed.

5.36 XY Group

:XY<x>?

Function Queries all XY display settings.

Syntax :XY<x>?

<x> = 1 or 2 (1 only for 2-channel models)

:XY<x>:DISPlay

Function Sets or queries whether to show (ON) or hide (OFF) the XY display on the screen.

Syntax :XY<x>:DISPlay {<Boolean>}

:XY<x>:DISPlay?

<x> = 1 or 2 (1 only for 2-channel models)

Example :XY1:DISPLAY ON

:XY1:DISPLAY?

-> :XY1:DISPLAY 1

:XY<x>:MEASure?

Function Queries all automated measurement settings of XY display.

Syntax :XY<x>:MEASure?

<x> = 1 or 2 (1 only for 2-channel models)

:XY<x>:MEASure:CURSor?

Function Queries all cursor measurement settings of XY display.

Syntax :XY<x>:MEASure:CURSor?

<x> = 1 or 2 (1 only for 2-channel models)

:XY<x>:MEASure:CURSor:DX?

Function Queries all settings for the voltage difference between the XY display's horizontal cursors.

Syntax :XY<x>:MEASure:CURSor:DX?

<x> = 1 or 2 (1 only for 2-channel models)

:XY<x>:MEASure:CURSor:DX:STATe

Function Sets or queries the on/off status of the voltage difference between the XY display's horizontal cursors.

Syntax :XY<x>:MEASure:CURSor:DX:STATe {<Boolean>}

:XY<x>:MEASure:CURSor:DX:STATe?

<x> = 1 or 2 (1 only for 2-channel models)

Example :XY1:MEASURE:CURSOR:DX:STATE ON

:XY1:MEASURE:CURSOR:DX:STATE?

-> :XY1:MEASURE:CURSOR:DX:STATE 1

:XY<x>:MEASure:CURSor:DX:VALue?

Function Queries the voltage difference between the XY display's horizontal cursors.

Syntax :XY<x>:MEASure:CURSor:DX:VALue?

<x> = 1 or 2 (1 only for 2-channel models)

Example :XY1:MEASURE:CURSOR:DX:VALUE?

-> :XY1:MEASURE:CURSOR:DX:VALUE 6.000E+00

:XY<x>:MEASure:CURSor:DY?

Function Queries all settings for the voltage difference between the XY display's vertical cursors.

Syntax :XY<x>:MEASure:CURSor:DY?

<x> = 1 or 2 (1 only for 2-channel models)

:XY<x>:MEASure:CURSor:DY:STATe

Function Sets or queries the on/off status of the voltage difference between the XY display's vertical cursors.

Syntax :XY<x>:MEASure:CURSor:DY:STATe {<Boolean>}

:XY<x>:MEASure:CURSor:DY:STATe?

<x> = 1 or 2 (1 only for 2-channel models)

Example :XY1:MEASURE:CURSOR:DY:STATE ON

:XY1:MEASURE:CURSOR:DY:STATE?

-> :XY1:MEASURE:CURSOR:DY:STATE 1

:XY<x>:MEASure:CURSor:DY:VALue?

Function Queries the voltage difference between the XY display's vertical cursors.

Syntax :XY<x>:MEASure:CURSor:DY:VALue?

<x> = 1 or 2 (1 only for 2-channel models)

Example :XY1:MEASURE:CURSOR:DY:VALUE?

-> :XY1:MEASURE:CURSOR:DY:VALUE 6.000E+00

:XY<x>:MEASure:CURSor:X<y>?

Function Queries all horizontal cursor settings of XY display.

Syntax :XY<x>:MEASure:CURSor:X<y>?

<x> = 1 or 2 (1 only for 2-channel models)

<y> = 1 or 2

:XY<x>:MEASure:CURSor:X<y>:POSition

Function Sets or queries a horizontal cursor position on the XY display.

Syntax
```
:XY<x>:MEASure:CURSor:X<y>:POSiti
on {<NRf>}
:XY<x>:MEASure:CURSor:X<y>:POSition?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = –4 div to 4 div

Example
```
:XY1:MEASURE:CURSOR:X1:POSITION 1
:XY1:MEASURE:CURSOR:X1:POSITION?
-> :XY1:MEASURE:CURSOR:X1:POSITION 1
.000E+00
```

:XY<x>:MEASure:CURSor:X<y>:STATe

Function Sets or queries the on/off status of the voltage of an XY display's horizontal cursor.

Syntax
```
:XY<x>:MEASure:CURSor:X<y>:STA
Te {<Boolean>}
:XY<x>:MEASure:CURSor:X<y>:STATe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:XY1:MEASURE:CURSOR:X1:STATE ON
:XY1:MEASURE:CURSOR:X1:STATE?
-> :XY1:MEASURE:CURSOR:X1:STATE 1
```

:XY<x>:MEASure:CURSor:X<y>:VALue?

Function Queries the voltage at a XY display's horizontal cursor.

Syntax
```
:XY<x>:MEASure:CURSor:X<y>:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:XY1:MEASURE:CURSOR:X1:VALUE?
-> :XY1:MEASURE:CURSOR:X1:VALUE 1.00
0E+00
```

:XY<x>:MEASure:CURSor:Y<y>?

Function Queries all vertical cursor settings of XY display.

Syntax
```
:XY<x>:MEASure:CURSor:Y<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

:XY<x>:MEASure:CURSor:Y<y>:POSition

Function Sets or queries a vertical cursor position on the XY display.

Syntax
```
:XY<x>:MEASure:CURSor:Y<y>:POSiti
on {<NRf>}
:XY<x>:MEASure:CURSor:Y<y>:POSition?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = –4 div to 4 div

Example
```
:XY1:MEASURE:CURSOR:Y1:POSITION 1
:XY1:MEASURE:CURSOR:Y1:POSITION?
-> :XY1:MEASURE:CURSOR:Y1:POSITION 1
.000E+00
```

:XY<x>:MEASure:CURSor:Y<y>:STATe

Function Sets or queries the on/off status of the voltage of an XY display's vertical cursor.

Syntax
```
:XY<x>:MEASure:CURSor:Y<y>:STA
Te {<Boolean>}
:XY<x>:MEASure:CURSor:Y<y>:STATe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:XY1:MEASURE:CURSOR:Y1:STATE ON
:XY1:MEASURE:CURSOR:Y1:STATE?
-> :XY1:MEASURE:CURSOR:Y1:STATE 1
```

:XY<x>:MEASure:CURSor:Y<y>:VALue?

Function Queries the voltage at a XY display's vertical cursor.

Syntax
```
:XY<x>:MEASure:CURSor:Y<y>:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:XY1:MEASURE:CURSOR:Y1:VALUE?
-> :XY1:MEASURE:CURSOR:Y1:VALUE 1.00
0E+00
```

:XY<x>:MEASure:INTeg?

Function Queries all integral settings of XY display.

Syntax
```
:XY<x>:MEASure:INTeg?
```
<x> = 1 or 2 (1 only for 2-channel models)

:XY<x>:MEASure:INTeg:LOOP

Function Sets or queries the integral method on the XY display.

Syntax
```
:XY<x>:MEASure:INTeg:LOOP {CLOSe|OPE
N}
:XY<x>:MEASure:INTeg:LOOP?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:XY1:MEASURE:INTEG:LOOP CLOSE
:XY1:MEASURE:INTEG:LOOP?
-> :XY1:MEASURE:INTEG:LOOP CLOSE
```

:XY<x>:MEASure:INTeg:POLarity

Function Sets or queries the integral direction on the XY display.

Syntax
```
:XY<x>:MEASure:INTeg:POLarity {CCW|CW}
:XY<x>:MEASure:INTeg:POLarity?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:XY1:MEASURE:INTEG:POLARITY CCW
:XY1:MEASURE:INTEG:POLARITY?
-> :XY1:MEASURE:INTEG:POLARITY CCW
```

:XY<x>:MEASure:INTeg:VALue?

Function Queries the integral value on the XY display.

Syntax
```
:XY<x>:MEASure:INTeg:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:XY1:MEASURE:INTEG:VALUE?
-> :XY1:MEASURE:INTEG:VALUE 1.000E+00
```

:XY<x>:MEASure:MODE

Function Sets or queries the automated measurement mode on the XY display.

Syntax
```
:XY<x>:MEASure:MODE {CURSor|INTeg|OFF}
:XY<x>:MEASure:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:XY1:MEASURE:MODE CURSOR
:XY1:MEASURE:MODE?
-> :XY1:MEASURE:MODE CURSOR
```

:XY<x>:RANGe

Function Sets or queries the VT waveform source window for the XY display.

Syntax
```
:XY<x>:RANGe{MAIN|Z1|Z2}
:XY<x>:RANGe?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:XY1:RANGE MAIN
:XY1:RANGE?
-> :XY1:RANGE MAIN
```

:XY<x>:SPLit

Function Sets or queries whether the XY1/XY2 display screen is split on the XY display.

Syntax
```
:XY<x>:SPLit {<Boolean>}
:XY<x>:SPLit?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:XY1:SPLIT ON
:XY1:SPLIT?
-> :XY1:SPLIT 1
```

:XY<x>:TRANge (Time Range)

Function Sets or queries the VT waveform range for the XY display.

Syntax
```
:XY<x>:TRANge {<NRf>,<NRf>}
:XY<x>:TRANge?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf>, <NRf> = –5 div to 5 div (in steps of 10 div/display record length)

Example
```
:XY1:TRANGE -4,4
:XY1:TRANGE?
-> :XY1:TRANGE -4.00,4.00
```

:XY<x>:VTDisplay

Function Sets or queries the on/off state of the VT waveform display on the XY display.

Syntax
```
:XY<x>:VTDisplay {<Boolean>}
:XY<x>:VTDisplay?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:XY1:VTDISPLAY ON
:XY1:VTDISPLAY?
-> :XY1:VTDISPLAY 1
```

:XY<x>:XTRace

Function Sets or queries the channel that is assigned to the X-axis on the XY display.

Syntax
```
:XY<x>:XTRace {<NRf>|MATH<y>}
:XY<x>:XTRace?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:XY1:XTRACE 1
:XY1:XTRACE?
-> :XY1:XTRACE 1
```

:XY<x>:YTRace

Function Sets or queries the channel that is assigned to the Y-axis on the XY display.

Syntax
```
:XY<x>:YTRace {<NRf>|MATH<y>}
:XY<x>:YTRace?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:XY1:YTRACE 1
:XY1:YTRACE?
-> :XY1:YTRACE 1
```

5.37 ZOOM Group

:ZOOM<x>?

Function Queries all waveform zoom settings.

Syntax :ZOOM<x>?

<x> = 1 or 2

:ZOOM<x>:ALLocation?

Function Queries all zoom source waveform settings.

Syntax :ZOOM<x>:ALLocation?

<x> = 1 or 2

:ZOOM<x>:ALLocation:{CHANnel<y>|MATH<z>}

Function Sets or queries the zoom source waveform.

Syntax :ZOOM<x>:ALLocation:{CHANnel<y>|MATH<z>} {<Boolean>}

:ZOOM<x>:ALLocation:{CHANnel<y>|MATH<z>}?

<x> = 1 or 2

<y> of CHANnel<y> = 1 to 4 (1 or 2 on 2-channel models)

<z> of MATH<z> = 1 to 4 (1 or 2 on 2-channel models)

Example :ZOOM1:ALLOCATION:CHANNEL1 ON

:ZOOM1:ALLOCATION:CHANNEL1?

-> :ZOOM1:ALLOCATION:CHANNEL1 1

:ZOOM<x>:DISPlay

Function Sets or queries the zoom waveform display on/off state.

Syntax :ZOOM<x>:DISPlay {<Boolean>}

:ZOOM<x>:DISPlay?

<x> = 1 or 2

Example :ZOOM1:DISPLAY ON

:ZOOM1:DISPLAY?

-> :ZOOM1:DISPLAY 1

:ZOOM<x>:FORMat

Function Sets or queries the zoom waveform display format.

Syntax :ZOOM<x>:FORMat {DUAL|HEXa|MAIN|OCTal|QUAD|SINGle|TRIad}

:ZOOM<x>:FORMat?

<x> = 1 or 2

Example :ZOOM1:FORMAT DUAL

:ZOOM1:FORMAT?

-> :ZOOM1:FORMAT DUAL

:ZOOM<x>:MAG

Function Sets or queries the magnification of a zoom waveform display window.

Syntax :ZOOM<x>:MAG {<NRf>}

:ZOOM<x>:MAG?

<x> = 1 or 2

<NRf> = See the Features Guide for this information.

Example :ZOOM1:MAG 2.5

:ZOOM1:MAG?

-> :ZOOM1:MAG 2.5

:ZOOM<x>:MAGFine

Function Sets or queries the magnification (fine) of a zoom waveform display window.

Syntax :ZOOM<x>:MAGFine {<NRf>}

:ZOOM<x>:MAGFine?

<x> = 1 or 2

<NRf> = See the Features Guide for this information.

Example :ZOOM1:MAGFINE 2.5

:ZOOM1:MAGFINE?

-> :ZOOM1:MAGFINE 2.5

:ZOOM<x>:MAIN

Function Sets or queries the zoom waveform display ratio.

Syntax :ZOOM<x>:MAIN {20|50|OFF}

:ZOOM<x>:MAIN?

<x> = 1 or 2

Example :ZOOM1:MAIN 20

:ZOOM1:MAIN?

-> :ZOOM1:MAIN 20

:ZOOM<x>:POSition

Function Sets or queries the position of a zoom box.

Syntax :ZOOM<x>:POSition {<NRf>}

:ZOOM<x>:POSition?

<x> = 1 or 2

<NRf> = –5 div to 5 div (in steps of 10 divisions/display record length)

Example :ZOOM1:POSITION 2

:ZOOM1:POSITION?

-> :ZOOM1:POSITION 2.00E+00

:ZOOM<x>:VERTical?

Function Queries all vertical zoom settings.

Syntax :ZOOM<x>:VERTical?

<x> = 1 or 2

:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}?

Function Queries all settings related to a vertical zoom trace.

Syntax
```
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}?
```
<x> = 1 or 2
<y> of CHANnel<y> = 1 to 4 (1 or 2 on 2-channel models)
<z> of MATH<z> = 1 to 4 (1 or 2 on 2-channel models)

:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}:MAG

Function Sets or queries the vertical zoom factor.

Syntax
```
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}:MAG {<NRf>}
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}:MAG?
```
<x> = 1 or 2
<y> of CHANnel<y> = 1 to 4 (1 or 2 on 2-channel models)
<z> of MATH<z> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = See the Features Guide for this information.

Example
```
:ZOOM1:VERTICAL:CHANNEL1:MAG 2.5
:ZOOM1:VERTICAL:CHANNEL1:MAG?
-> :ZOOM1:VERTICAL:CHANNEL1:MAG 2.5
```

:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}:POSition

Function Sets or queries the vertical zoom position.

Syntax
```
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}:POSition {<NRf>}
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<z>}:POSition?
```
<x> = 1 or 2
<y> of CHANnel<y> = 1 to 4 (1 or 2 on 2-channel models)
<z> of MATH<z> = 1 to 4 (1 or 2 on 2-channel models)
<NRf> = –4 div to 4 div

Example
```
:ZOOM1:VERTICAL:CHANNEL1:POSITION 2
:ZOOM1:VERTICAL:CHANNEL1:POSITION?
-> :ZOOM1:VERTICAL:CHANNEL1:
POSITION 2.00E+00
```

:ZOOM<x>:VERTical:TRACe

Function Sets or queries the trace to show on the vertical zoom window.

Syntax
```
:ZOOM<x>:VERTical:TRACe {<NRf>|MATH<y>}
:ZOOM<x>:VERTical:TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 4 (1 or 2 on 2-channel models)
<y> of MATH<y> = 1 to 4 (1 or 2 on 2-channel models)

Example
```
:ZOOM1:VERTICAL:TRACE 1
:ZOOM1:VERTICAL:TRACE?
-> :ZOOM1:VERTICAL:TRACE 1
```

5.38 Common Command Group

The commands in this group are defined in USBTMC-USB488 and are independent from the instrument-specific functions. There are no front panel keys that correspond to the commands in this group.

*CAL? (CALibrate)

Function Executes calibration and queries the result.
Syntax `*CAL?`
Example `*CAL?`
`-> 0`
Description If the calibration ends normally, 0 is returned. If an error is detected, 1 is returned.

*CLS (CLear Status)

Function Clears the standard event register, extended event register, and error queue.
Syntax `*CLS`
Example `*CLS`
Description
- If the *CLS command is located immediately after the program message terminator, the output queue is also cleared.
- For information about each register and queue, see chapter 6.

*ESE (standard Event Status Enable register)

Function Sets or queries the standard event enable register.
Syntax `*ESE {<NRf>}`
`*ESE?`
<NRf> = 0 to 255
Example `*ESE 251`
`*ESE?`
`-> 251`
Description
- Specify the value as a sum of the values of each bit in decimal format.
- For example, specifying *ESE 251 will cause the standard enable register to be set to 11111011. In this case, bit 2 of the standard event register is disabled. This means that bit 5 (ESB) of the status byte register is not set to 1, even if a query error occurs.
- The default value is *ESE 0 (all bits disabled).
- A query using *ESE? will not clear the contents of the standard event enable register.
- For information about the standard event enable register, see page 6-4.

*ESR? (standard Event Status Register)

Function Queries and clears the standard event register.
Syntax `*ESR?`
Example `*ESR?`
`-> 32`
Description
- A sum of the values of each bit is returned in decimal format.
- When an SRQ is sent, you can check what types of events have occurred.
- For example, if a value of 32 is returned, this indicates that the standard event register is set to 00100000. This means that the SRQ occurred due to a command syntax error.
- A query using *ESR? will clear the contents of the Standard Event Register.
- For information about the standard event register, see page 6-4.

*IDN? (IDeNtify)

Function Queries the instrument model.
Syntax `*IDN?`
Example `*IDN?`
`-> YOKOGAWA,710120,08ELP1024,F1.01`
Description This command returns a string in the following format: <Manufacturer>, <Model>, <Serial no.>, <Firmware version>. For <model>, the instrument returns “DLM3022,” “DLM3024,” “DLM3032,” “DLM3034,” “DLM3052,” and “DLM3054” for DLM3022, DLM3024, DLM3032, DLM3034, DLM3052 and DLM3054, respectively.

*OPC (OPeration Complete)

Function Sets bit 0 (the OPC bit) of the standard event register to 1 upon the completion of the specified overlap command.
Syntax `*OPC`
Example `*OPC`
Description
- For information about how to synchronize a program using *OPC, see page 4-8.
- The COMMunicate:OPSE command is used to specify the overlap command.
- If *OPC is not the last command of the message, its operation is not guaranteed.

*OPC? (OPeration Complete)

Function When you send *OPC?, the instrument returns ASCII code 1 when the specified overlap command is completed.

Syntax
```
*OPC?
```

Example
```
*OPC?
-> 1
```

Description
- For information about how to synchronize a program using *OPC?, see page 4-8.
- The COMMunicate:OPSE command is used to specify the overlap command.
- If *OPC? is not the last command of the message, its operation is not guaranteed.

*OPT? (OPTion)

Function Queries the installed options.

Syntax
```
*OPT?
```

Example
```
*OPT?
-> 500MPOINTS,LOGIC,PROBEPOWER4,GPIB
,STORAGE,USERDEFINE,PANALYZE,UART,
I2C,SPI,CAN,CAN FD,LIN,FLEXRAY,SENT,
CXPI
```

Description
- The instrument returns the availabilities of <memory model>, <switch logic>, <printer>, <rear-panel probe power>, <GP-IB>, <internal storage>, <user-defined computation>, <power supply analysis feature>, <UART analysis feature>, <I2C analysis feature>, <SPI analysis feature>, <CAN analysis feature>, <CAN FD analysis feature>, <LIN analysis feature>, <FlexRay analysis feature>, <CXPI analysis feature>, and so on.
- *OPT? must be the last query in the program message. An error occurs if there is a query after this query.

*RST (ReSeT)

Function Initializes the settings.

Syntax
```
*RST
```

Example
```
*RST
```

Description Also clears *OPC and *OPC? commands that have been sent.

*SRE (Service Request Enable register)

Function Sets or queries the service request enable register value.

Syntax
```
*SRE <NRf>
*SRE?
```
<NRf> = 0 to 255

Example
```
*SRE 239
*SRE?
-> 175
```

Description
- Specify the value as a sum of the values of each bit in decimal format.
- For example, specifying *SRE 239 will cause the standard enable register to be set to 11101111. In this case, bit 4 of the service request enable register is disabled. This means that bit 4 (MAV) of the status byte register is not set to 1, even if the output queue is not empty.
- Bit 6 (MSS) of the status byte register is the MSS bit itself and is therefore ignored.
- The default value is *SRE 0 (all bits disabled).
- A query using *SRE? will not clear the contents of the service request enable register.
- For information about the service request enable register, see page 6-2.

*STB? (STatus Byte)

Function Queries the status byte register value.

Syntax
```
*STB?
```

Example
```
*STB?
-> 4
```

Description
- A sum of the values of each bit is returned as a decimal value.
- Because the register is read without executing serial polling, bit 6 is an MSS bit, not an RQS bit.
- For example, if a value of 4 is returned, this indicates that the status byte register is set to 00000100. This means that the error queue is not empty (in other words, an error occurred).
- A query using *STB? will not clear the contents of the status byte register.
- For information about the status byte register, see page 6-2.

***TST?**

Function Executes a self-test and queries the result. The self-test consists of tests of each kind of internal memory.

Syntax `*TST?`

Example `*TST?`
`-> 0`

Description This command returns 0 if the self-test is successful and 1 if it is not.

***WAI (WAIt)**

Function Holds the execution of the subsequent command until the specified overlap command is completed.

Syntax `*WAI`

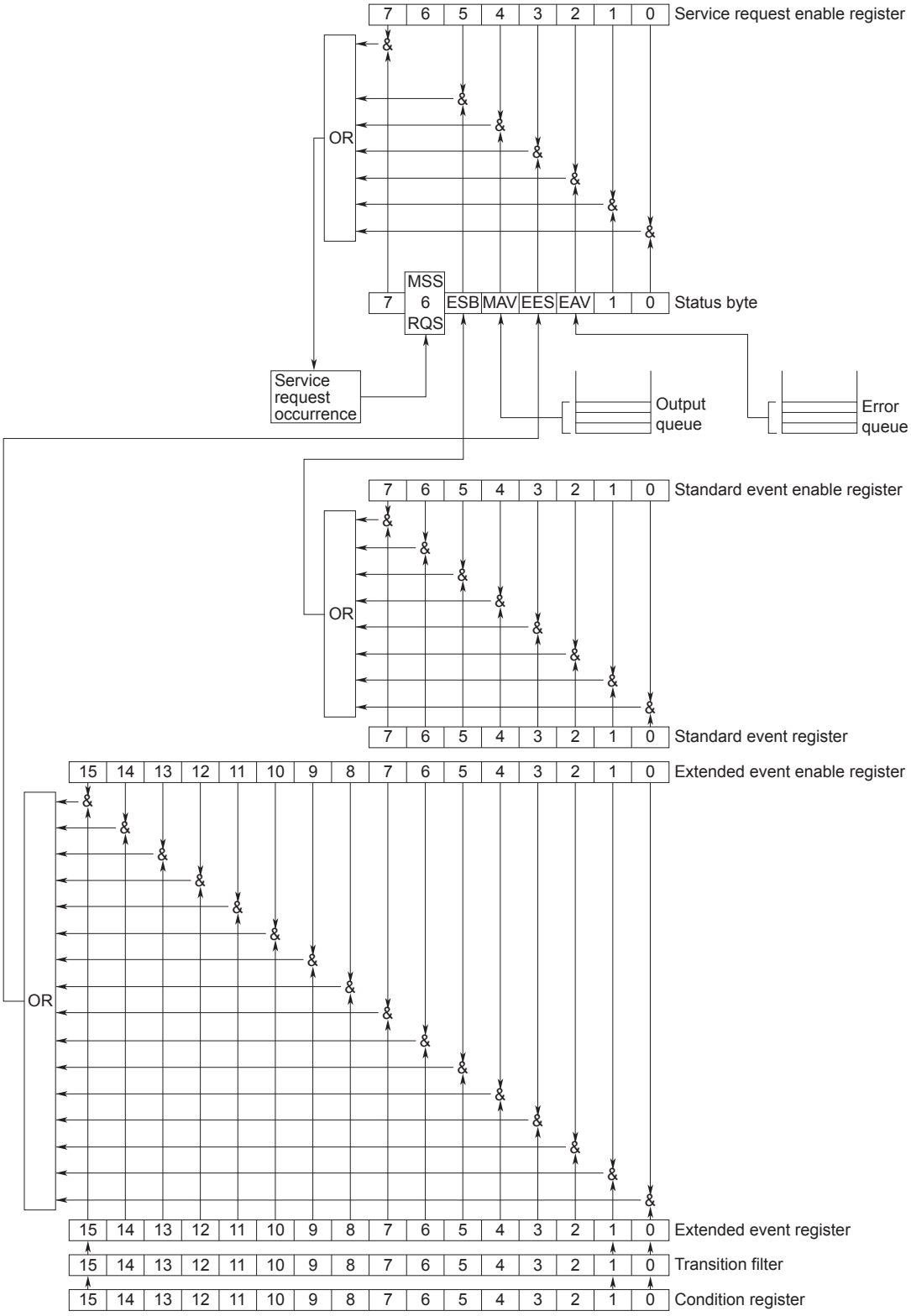
Example `*WAI`

Description
- For information about how to synchronize a program using *WAI, see page 4-8.
- The COMMunicate:OPSE command is used to specify the overlap command.

6.1 About Status Reports

Status Reports

The figure below shows the format of status reports that are read by serial polling. This status report format is an extended version of the status report format defined in IEEE 488.2-1992.



Overview of Registers and Queues

Name	Function	Write	Read
Status byte	–	–	Serial polling `(RQS)`, `*STB? (MSS)`
Service request enable register	Status byte mask	`*SRE`	`*SRE?`
Standard event register	Indicates device status changes	–	`*ESR?`
Standard event enable register	Standard event register mask	`*ESE`	`*ESE?`
Extended event register	Indicates device status changes	–	`STATus:EESR?`
Extended event enable register	Extended event register mask	`STATus:EESE`	`STATus:EESE?`
Condition register	Current device status	–	`STATus:CONDition?`
Transition filter	Conditions that change the extended event register	`STATus:FILTer<x>`	`STATus:FILTer<x>?`
Output queue	Stores response messages for queries	Query commands	
Error queue	Stores error numbers and messages	–	`STATus:ERRor?`

Registers and Queues That Affect the Status Byte

The following registers affect the status byte bits.

Standard event register	Sets bit 5 (ESB) of the status byte to 1 or 0.
Output queue	Sets bit 4 (MAV) of the status byte to 1 or 0.
Extended event register	Sets bit 3 (EES) of the status byte to 1 or 0.
Error queue	Sets bit 2 (EAV) of the statys byte to 1 or 0.

Enable Registers

The following registers are used to mask a bit so that the bit will not affect the status byte even when it is set to 1.

Service request enable register	Masks bits of the status byte.
Standard event enable register	Masks bits in the standard event register.
Extended event enable register	Masks bits in the extended event register.

Reading and Writing to Registers

For example, you can use the `*ESE` command to set the standard event enable register bits to ones and zeros. You can use the `*ESE?` command to query whether the standard event enable register bits are ones or zeros. For details on commands, see chapter 5.

6.2 Status Byte

Status byte

7	RQS 6 MSS	ESB	MAV	EES	EAV	1	0

- **Bits 0, 1, and 7**
  Not used (always zero)
- **Bit 2 EAV (Error Available)**
  This bit is set to 1 when the error queue is not empty. In other words, this bit is set to 1 when an error occurs. For details, see page 6-6.
- **Bit 3 EES (Extend Event Summary Bit)**
  This bit is set to 1 when the logical AND of the extended event register and its corresponding event register is 1. In other words, this bit is set to 1 when a certain event takes place inside the instrument. For details, see page 6-5.
- **Bit 4 MAV (Message Available)**
  This bit is set to 1 when the output queue is not empty. In other words, this bit is set to 1 when there is data to be transmitted in response to a query. For details, see page 6-6.
- **Bit 5 ESB (Event Summary Bit)**
  This bit is set to 1 when the logical AND of the standard event register and its corresponding event register is 1. In other words, this bit is set to 1 when a certain event takes place inside the instrument. For details, see page 6-4.
- **Bit 6 RQS (Request Service)/MSS (Master Status Summary)**
  This bit is set to 1 when the logical AND of the status byte excluding bit 6 and the service request enable register is 1. In other words, this bit is set to 1 when the instrument is requesting service from the controller.
  RQS is set to 1 when the MSS bit changes from 0 to 1 and is cleared when serial polling is carried out or when the MSS bit changes to 0.

Bit Masking

To mask a bit in the status byte so that it does not trigger an SRQ, set the corresponding bit of the service request enable register to zero.
For example, to mask bit 2 (EAV) so that service is not requested when an error occurs, set bit 2 of the service request enable register to 0. Do this using the `*SRE` command. To query whether each bit of the service request enable register is 1 or 0, use `*SRE?`. For details on the `*SRE` command, see chapter 5.

Status Byte Operation

A service request is issued when bit 6 in the status byte becomes 1. Bit 6 is set to 1 when any other bit is 1 (when the corresponding bit of the service request enable register is also set to 1). For example, if an event occurs and the logical AND of a standard event register bit and its corresponding enable register bit is 1, then bit 5 (ESB) is set to 1. At this point, if bit 5 of the service request enable register is 1, bit 6 (MSS) is set to 1, and the instrument requests service from the controller.
You can check what type of event occurred by reading the contents of the status byte.

Reading the Status Byte

There are two ways to read the contents of the status byte.

- ***STB? Query**
  An `*STB?` query causes bit 6 to function as an MSS bit. This query does not cause any of the status byte bits to be cleared after the status byte is read.
- **Serial Polling**
  Serial polling causes bit 6 to function as an RQS bit. After the status byte is read, only the RQS bit is cleared. You cannot read the MSS bit when serial polling is used.

Clearing the Status Byte

There is no way to clear all of the bits in the status byte. The bits that are cleared vary for each operation as follows:

- ***STB? Query**
  None of the bits are cleared.
- **Serial Polling**
  Only the RQS bit is cleared.
- **When a *CLS command is received**
  When a `*CLS` command is received, the status byte itself is not cleared, but the contents of the standard event register, which affect the bits in the status byte, are cleared. As a result, the corresponding status byte bits are cleared. Because the output queue is not cleared with a `*CLS` command, bit 4 (MAV) in the status byte is not affected. However, the output queue will be cleared if the `*CLS` command is received just after a program message terminator.

6.3 Standard Event Register

Standard Event Register

7	6	5	4	3	2	1	0
PON	URQ	CME	EXE	DDE	QYE	RQC	OPC

- **Bit 7 PON (Power ON)**
  This bit is set to 1 when the instrument is turned on.
- **Bit 6 URQ (User Request)**
  Not used (always zero)
- **Bit 5 CME (Command Error)**
  This bit is set to 1 when there is a command syntax error.
  Example Incorrectly spelled command name; 9 used in octal data.
- **Bit 4 EXE (Execution Error)**
  This bit is set to 1 when the command syntax is correct, but the command cannot be executed in the current state.
  Example The instrument receives a command whose parameter is outside the selectable range. An attempt is made to print a hard copy while the instrument is running.
- **Bit 3 DDE (Device Error)**
  This bit is set to 1 when a command cannot be executed for internal reasons other than a command syntax error or command execution error.
- **Bit 2 QYE (Query Error)**
  This bit is set to 1 when a query command is received, but the output queue is empty or the data is lost.
  Example There is no response data. Data is lost due to an overflow in the output queue.
- **Bit 1 RQC (Request Control)**
  Not used (always zero)
- **Bit 0 OPC (Operation Complete)**
  This bit is set to 1 upon the completion of the operation designated by the `*OPC` command (see chapter 5 for details).

Bit Masking

To mask a certain bit of the standard event register so that it does not cause bit 5 (ESB) in the status byte to change, set the corresponding bit of the standard event enable register to zero.
For example, to mask bit 2 (QYE) so that ESB will not be set to 1 even if a query error occurs, set bit 2 of the standard event enable register to zero. Do this using the `*ESE` command. To query whether each bit of the standard event enable register is 1 or 0, use `*ESE?`.
For details on the `*ESE` command, see chapter 5.

Standard Event Register Operation

The standard event register indicates eight types of events that occur inside the instrument. When one of the bits in this register is 1 (and the corresponding bit of the standard event enable register is also 1), bit 5 (ESB) in the status byte is set to 1 .
Example
1. A query error occurs.
2. Bit 2 (QYE) is set to 1.
3. When bit 2 of the standard event enable register is 1, bit 5 (ESB) in the status byte is set to 1.

You can also check what type of event occurred in the instrument by reading the contents of the standard event register.

Reading the Standard Event Register

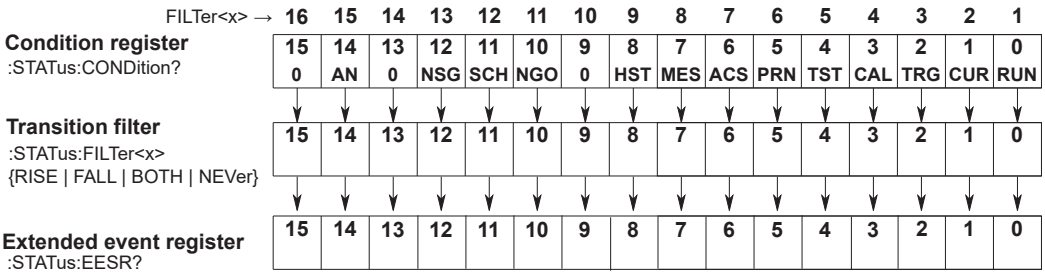
You can use the `*ESR?` command to read the contents of the standard event register. The register is cleared after it is read.

Clearing the Standard Event Register

The standard event register is cleared when:
- The contents of the standard event register are read using the `*ESR?` command.
- A `*CLS` command is received.
- The instrument is turned off and then back on.

6.4 Extended Event Register

The extended event register receives information about changes in the condition register, which indicates the instrument's internal condition. The information is the result of edge detection performed by the transition filter.



The condition register bits are described below.

Bit 0	RUN (Running)	This bit is set to 1 when waveform acquisition is in progress.
Bit 1	CUR (Cursor)	This bit is set to 1 when cursor measurement is in progress.
Bit 2	TRG (Awaiting trigger)	This bit is set to 1 when the instrument is waiting for a trigger.
Bit 3	CAL (Calibration)	This bit is set to 1 when calibration is in progress.
Bit 4	TST (Testing)	This bit is set to 1 when a self-test is in progress.
Bit 5	PRN (Printing)	This bit is set to 1 when the built-in printer is in operation, when data is being transmitted to an external USB or network printer, or when screen capture data is being saved.
Bit 6	ACS (Accessing)	This bit is set to 1 when a drive is being accessed.
Bit 7	MES (Measuring)	This bit is set to 1 when automated measurement of waveform parameters is in progress.
Bit 8	HST (History Search)	This bit is set to 1 when a history search is in progress.
Bit 10	NGO (Go/No-go)	This bit is set to 1 when a GO/NO-GO search is in progress.
Bit 11	SCH (Search)	This bit is set to 1 when a waveform search is in progress (this includes when a serial bus signal search is in progress).
Bit 12	NSG (N-Single)	This bit is set to 1 when consecutive acquisition is in progress when the trigger mode is set to Single (N).
Bit 14	AN (Analysis)	This bit is set to 1 when an analysis is in progress (this includes when a serial bus signal analysis is in progress).

The transition filter parameters detect changes in the specified condition register bits (numeric suffixes 1 to 16) and overwrite the extended event register in the following ways.

RISE	The specified extended event register bit is set to 1 when the corresponding condition register bit changes from 0 to 1.
FALL	The specified extended event register bit is set to 1 when the corresponding condition register bit changes from 1 to 0.
BOTH	The specified extended event register bit is set to 1 when the corresponding condition register bit changes from 0 to 1 or from 1 to 0.
NEVer	Always zero.

6.5 Output and Error Queues

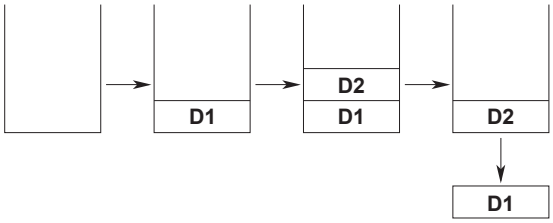
Output Queue

The output queue stores query response messages. For example, if you send a `:WAVeform:SEND?` command, which requests for the transmission of acquired data, the data is stored in the output queue until it is read.

As shown below, data is stored in order and read from the oldest message first. The output queue is cleared when:

- A new message is received from the controller.
- A deadlock occurs (see page 4-2).
- A device clear command (DCL or SDC) is received.
- The instrument is turned off and then back on.

The `*CLS` command does not clear the output queue. You can determine whether or not the output queue is empty by checking the status byte bit 4 (MAV).



Error Queue

When an error occurs, the error queue stores the error number and message. For example, if the instrument receives an incorrect program message from the controller, the error number (113) and the error message (“Undefined header”) are stored in the error queue when the instrument displays the error message.

You can use the `:STATus:ERRor?` query to read the contents of the error queue. Like the output queue, the messages in the error queue are read from the oldest one first.

If the error queue overflows, the last message is replaced with the following message: 350, “Queue overflow”

The error queue is cleared when:

- A `*CLS` command is received.
- The instrument is turned off and then back on.

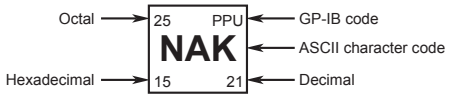
You can determine whether or not the error queue is empty by checking bit 2 in the status byte (EAV).

Appendix 1 ASCII Character Codes

The following table contains ASCII character codes.

	0	1	2	3	4	5	6	7
0	0 **NUL** 0 0	20 **DEL** 10 16	40 0 **SP** 20 32	60 16 **0** 30 48	100 0 **@** 40 64	120 16 **P** 50 80	140 0 **‘** 60 96	160 16 **p** 70 112
1	1 GTL **SOH** 1 1	21 LLO **DC1** 11 17	41 1 **!** 21 33	61 17 **1** 31 49	101 1 **A** 41 65	121 17 **Q** 51 81	141 1 **a** 61 97	161 17 **q** 71 113
2	2 **STX** 2 2	22 **DC2** 12 18	42 2 **”** 22 34	62 18 **2** 32 50	102 2 **B** 42 66	122 18 **R** 52 82	142 2 **b** 62 98	162 18 **r** 72 114
3	3 **ETX** 3 3	23 **DC3** 13 19	43 3 **#** 23 35	63 19 **3** 33 51	103 3 **C** 43 67	123 19 **S** 53 83	143 3 **c** 63 99	163 19 **s** 73 115
4	4 SDC **EOT** 4 4	24 DCL **DC4** 14 20	44 4 **$** 24 36	64 20 **4** 34 52	104 4 **D** 44 68	124 20 **T** 54 84	144 4 **d** 64 100	164 20 **t** 74 116
5	5 PPC **ENQ** 5 5	25 PPU **NAK** 15 21	45 5 **%** 25 37	65 21 **5** 35 53	105 5 **E** 45 69	125 21 **U** 55 85	145 5 **e** 65 101	165 21 **u** 75 117
6	6 **ACK** 6 6	26 **SYN** 16 22	46 6 **&** 26 38	66 22 **6** 36 54	106 6 **F** 46 70	126 22 **V** 56 86	146 6 **f** 66 102	166 22 **v** 76 118
7	7 **BEL** 7 7	27 **ETB** 17 23	47 7 **’** 27 39	67 23 **7** 37 55	107 7 **G** 47 71	127 23 **W** 57 87	147 7 **g** 67 103	167 23 **w** 77 119
8	10 GET **BS** 8 8	30 SPE **CAN** 18 24	50 8 **(** 28 40	70 24 **8** 38 56	110 8 **H** 48 72	130 24 **X** 58 88	150 8 **h** 68 104	170 24 **x** 78 120
9	11 TCT **HT** 9 9	31 SPD **EM** 19 25	51 9 **)** 29 41	71 25 **9** 39 57	111 9 **I** 49 73	131 25 **Y** 59 89	151 9 **i** 69 105	171 25 **y** 79 121
A	12 **LF** A 10	32 **SUB** 1A 26	52 10 ***** 2A 42	72 26 **:** 3A 58	112 10 **J** 4A 74	132 26 **Z** 5A 90	152 10 **j** 6A 106	172 26 **z** 7A 122
B	13 **VT** B 11	33 **ESC** 1B 27	53 11 **+** 2B 43	73 27 **;** 3B 59	113 11 **K** 4B 75	133 27 **[** 5B 91	153 11 **k** 6B 107	173 27 **{** 7B 123
C	14 **FF** C 12	34 **FS** 1C 28	54 12 **,** 2C 44	74 28 **<** 3C 60	114 12 **L** 4C 76	134 28 **** 5C 92	154 12 **l** 6C 108	174 28 **|** 7C 124
D	15 **CR** D 13	35 **GS** 1D 29	55 13 **-** 2D 45	75 29 **=** 3D 61	115 13 **M** 4D 77	135 29 **]** 5D 93	155 13 **m** 6D 109	175 29 **}** 7D 125
E	16 **SO** E 14	36 **RS** 1E 30	56 14 **.** 2E 46	76 30 **>** 3E 62	116 14 **N** 4E 78	136 30 **^** 5E 94	156 14 **n** 6E 110	176 30 **~** 7E 126
F	17 **SI** F 15	37 **US** 1F 31	57 15 **/** 2F 47	77 UNL **?** 3F 63	117 15 **O** 4F 79	137 UNT **_** 5F 95	157 15 **o** 6F 111	177 **DEL (RUBOUT)** 7F 127
	Address command	Universal command	Listener address		Talker address		Secondary command	

Example



Appendix 2 Error Messages

This section explains communication error messages.

- Messages can be displayed in English or in another language on the instrument. However, when they are read from a PC or other similar device, messages are displayed in English.
- If servicing is necessary to solve the problem indicated by a message, contact your nearest YOKOGAWA dealer.
- Only communication error messages are listed here. For other error messages, see the *User's Manual IM DLM3054-03EN*.
  - Communication syntax errors 100 to 199
  - Communication execution errors 200 to 299
  - Device-specific and other errors 300 to 399
  - Communication query errors 400 to 499
  - System error (communication) 399

  } Listed below

Communication Syntax Errors (100 to 199)

Code	Message	Corrective Action	Page
100	Command error.	Check the spelling of the command.	Chapter 5
102	Syntax error.	A syntax error not covered by error codes 100 to 199.	Chapters 4 and 5
103	Invalid separator.	Separate data values with a comma.	4-1
104	Data type error.	Use the correct data type for each parameter.	4-6 and 4-7
105	GET not allowed.	GET is not supported as a response to an interface message.	3-7
108	Parameter not allowed.	Check the number of data values.	4-6 and chapter 5
109	Missing parameter.	Be sure to include all necessary data values.	4-6 and chapter 5
111	Header separator error.	Use a comma to separate each header from its data.	4-1
112	Program mnemonic too long.	Check the command length.	Chapter 5
113	Undefined header.	Check the header.	4-4 and chapter 5
114	Header suffix out of range.	Check the header.	4-4 and chapter 5
120	Numeric data error.	A value must be specified where the syntax contains <NRf>.	4-6
123	Exponent too large.	Where the syntax contains <NR3>, make the exponent that follows E smaller.	4-6 and chapter 5
124	Too many digits.	Limit numeric values to 255 digits or less.	4-6 and chapter 5
128	Numeric data not allowed.	Use a data type other than <NRf>.	4-6 and chapter 5
131	Invalid suffix.	Check the units where the syntax contains <Voltage>, <Time>, <Frequency>, or <Current>.	4-6
134	Suffix too long.	Check the units where the syntax contains <Voltage>, <Time>, <Frequency>, or <Current>.	4-6
138	Suffix not allowed.	Units of measurement can only be used where the syntax contains <Voltage>, <Time>, <Frequency>, or <Current>.	4-6
141	Invalid character data.	Be sure to select one of the listed choices when the syntax contains {...|...|...}.	4-7 and chapter 5
144	Character data too long.	Check the spelling of the strings when the syntax contains {...|...|...}.	4-7 and chapter 5
148	Character data not allowed.	Use a data type other than <String data>.	4-6 and chapter 5
150	String data error.	Enclose parameters with single or double quotation marks where the syntax contains <String data>.	4-7

Code	Message	Corrective Action	Page
151	Invalid string data.	The parameter is either too long, or it contains an unusable character.	4-7 and chapter 5
158	String data not allowed.	Use a data type other than <String data>.	4-6 and chapter 5
161	Invalid block data.	<Block data> cannot be used.	4-7 and chapter 5
168	Block data not allowed.	<Block data> cannot be used.	4-7 and chapter 5
171	Missing Right.	Mathematical operations cannot be used.	—
172	Invalid expression.	Mathematical operations cannot be used.	Chapter 5
178	Expression data not allowed.	Mathematical operations cannot be used.	Chapter 5
181	Invalid outside macro definition.	The instrument does not support the IEEE488.2 macro specifications.	—

Communication Execution Errors (200 to 299)

Code	Message	Corrective Action	Page
221	Setting conflict.	Check settings that are related to each other.	Chapter 5
222	Data out of range.	Check the ranges of the settings.	Chapter 5
223	Too much data.	Check data byte lengths.	Chapter 5
224	Illegal parameter value.	Check the ranges of the settings.	Chapter 5
225	OverFlow.	Keep program messages to 1024 bytes or less in length, including <PMT>.	4-2
226	Out Of Memory.	Keep program messages to 1024 bytes or less in length, including <PMT>.	4-2
241	Hardware missing.	Check that the specified options are all installed.	—
260	Expression error.	Mathematical operations cannot be used.	—
270	Macro error.	The instrument does not support the IEEE488.2 macro specifications.	—
272	Macro execution error.	The instrument does not support the IEEE488.2 macro specifications.	—
273	Illegal macro label.	The instrument does not support the IEEE488.2 macro specifications.	—
275	Macro definition too long.	The instrument does not support the IEEE488.2 macro specifications.	—
276	Macro recursion error.	The instrument does not support the IEEE488.2 macro specifications.	—
277	Macro redefinition not allowed.	The instrument does not support the IEEE488.2 macro specifications.	—
278	Macro header not found.	The instrument does not support the IEEE488.2 macro specifications.	—

Communication Query Errors (400 to 499)

Code	Message	Corrective Action	Page
410	Query INTERRUPTED.	Check the transmission and reception order.	4-2
420	Query UNTERMINATED.	Check the transmission and reception order.	4-2
430	Query DEADLOCKED.	Keep program messages to 1024 bytes or less in length, including <PMT>.	4-2
440	Query UNTERMINATED after indefinite response.	Do not write a query after *IDN? or *OPT?.	—

System Communication Errors (300 and 399)
Error in System Operation

Code	Message	Corrective Action	Page
300	Communication device-specific error.	Servicing required.	—
399	Fatal error in the communication driver.	Servicing required.	—

Communication Warning (1)

Code	Message	Corrective Action	Page
1	*OPC/? exists in message.	Write *OPC or *OPC? at the end of program messages.	—

Other Error (350 and 390)

Code	Message	Corrective Action	Page
350	Queue overflow.	Read the error queue.	6-6
390	Communication overrun error.	Reduce the baud rate.	—

Note

Code 350 occurs when the error queue overflows. This error is only returned in response to a `:STATus:ERRor?` query; it is never displayed on the screen.

Appendix 3 About the IEEE 488.2-1992 Standard

The instrument's GP-IB interface conforms to the IEEE 488.2-1992 standard. This standard specifies that the following 23 items be stated in the document. This section describes these items.

(1) Of the IEEE 488.1 interface functions, the subsets that are supported
See section 3.4, “GP-IB Interface Specifications.”

(2) The operation of the device when it is assigned an address outside the 0 to 30 range
The address of this instrument cannot be set to an address outside the 0 to 30 range.

(3) Reaction of the device when the user changes the address
The address change is detected when the user presses UTIL and then the Remote Control soft key, and changes the address. The new address is valid until the next time it is changed.

(4) Device settings at power-on. The commands that can be used at power-on.
As a basic rule, the previous settings (the settings that were in use when the instrument seies was turned off) are used.
There are no limitations on the commands that can be used at power-on.

(5) Message exchange options

(a) Input buffer size
16384 bytes.

(b) Queries that return multiple response messages
See the example of the commands given in chapter 5.

(c) Queries that create response data when the command syntax is being analyzed
All queries create response data when the command syntax is analyzed.

(d) Queries that create response data during reception
There are no queries of which the response data are created upon receiving a send request from the controller.

(e) Commands that have parameters that restrict one another
There are commands such as :CHANnel<x>:PROBe[:MODE] and CHANnel<x>:VDIV that have parameters that place restrictions unilaterally, but there are no commands that have parameters that restrict one another.

(6) Items that are included in the functional or composite header elements constituting a command
See chapters 4 and 5.

(7) Buffer sizes that affect block data transmission
When block data is being transmitted, the output queue is expanded to match the size of the data that is being transmitted.

(8) A list of program data elements that can be used in equations and their nesting limitations
Equations cannot be used.

(9) Syntax of the responses to queries
See the example of the commands given in chapter 5.

(10) Communication between devices that do not follow the response syntax is not supported by the instrument series.

(11) Size of the response data block
1 to 999999998 bytes

(12) A list of supported common commands
See section 5.38, “Common Command Group.”

(13) Device condition after a successful calibration
The device will be performing measurements.

(14) The maximum length of block data that can be used for the *DDT trigger macro definition
Not supported.

(15) The maximum length of the macro label for defining macros, the maximum length of block data that can be used for the macro definition, and the process when recursion is used in macro definitions
Macro functions are not supported.

(16) Reply to the *IDN? query
See section 5.38, “Common Command Group.”

(17) Size of storage area for protected user data for PUD and *PUD?
*PUD and *PUD? are not supported.

(18) The length of the *RDT and *RDT? resource names
*RDT and *RDT? are not supported.

(19) The change in the status due to *RST, *LRN?, *RCL, and *SAV

*RST

See section 5.38, “Common Command Group.”

*LRN?, *RCL and *SAV

These common commands are not supported.

(20) The extent of the self-test using the *TST? command

Performs the same internal memory test that is executed when the user presses UTIL and then the Self Test soft key, and executes the MEMORY test.

(21) The structure of the extended return status

See chapter 6.

(22) Whether each command is processed in an overlapped manner or sequentially

See section 4.5, “Synchronization with the Controller” and chapter 5.

(23) The description of the execution of each command

See the explanations of each command’s function in chapter 5; the *Features Guide, IM DLM3054-01EN*; and the *User’s Manual, IM DLM3054-02EN*.

Index

Symbols

A

B

C

D

E

Q

R

S

T

U

V

W

X

Z