DLM2000 Series
Digital Oscilloscope
Mixed Signal Oscilloscope
Communication Interface

USER'S MANUAL

YOKOGAWA
Yokogawa Test & Measurement Corporation

IM 710105-17E
11th Edition

Thank you for purchasing the DLM2000 Series Digital Oscilloscope/Mixed Signal Oscilloscope. This Communication Interface User's Manual explains the following interface features and commands.

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

To ensure correct use, please read this manual thoroughly before operation.

Keep this manual in a safe place for quick reference in the event that a question arises.

List of Manuals

The following four manuals, including this one, are provided as manuals for the DLM2000. Read them along with this manual.

Manual Title	Manual No.	Description
DLM2000 Series Mixed Signal Oscilloscope Features Guide	IM 710105-01E	The supplied CD contains the PDF file of this manual. The manual explains the DLM2000 features.
DLM2000 Series Mixed Signal Oscilloscope User's Manual	IM 710105-02E	The supplied CD contains the PDF file of this manual. The manual explains how to operate the DLM2000.
DLM2000 Series Mixed Signal Oscilloscope Operation Guide	IM 710105-03E	This guide explains the handling precautions and basic operations of the DLM2000.
DLM2000 Series Mixed Signal Oscilloscope Communication Interface User's Manual	IM 710105-17E	This manual. The supplied CD contains the PDF file of this manual. The manual explains the DLM2000 communication interface features and instructions on how to use them.
DLM2000 Series Digital Oscilloscope Mixed Signal Oscilloscope	IM 710105-92	Document for China

The "E" in the manual number is the language code.

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 functionality. 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.

11th Edition: October 2017 (YMI)

Trademark

- 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.

About the USB Interface and Ethernet Interface

- To use the USB communication features, your PC must have the following:
  - DLM Series Library (TMCTL)
  - USB device driver for connecting the DLM2000 series to the PC
- To use the Ethernet communication features, your PC must have the following:
  - DLM Series Library (TMCTL)

  To download the libraries and drivers listed above, go to the following Web page, and then browse to the download page.
  http://tmi.yokogawa.com/service-support/downloads/

Sample Programs

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

http://tmi.yokogawa.com/service-support/downloads/

Revisions

• 1st Edition:	November 2008
• 2nd Edition:	March 2009
• 3rd Edition:	February 2010
• 4th Edition:	April 2011
• 5th Edition:	July 2012
• 6th Edition:	December 2013
• 7th Edition:	August 2014
• 8th Edition:	May 2015
• 9th Edition:	January 2016
• 10th Edition:	October 2016
• 11th Edition:	October 2017

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 (Optional)

Describes the features and specifications of the Ethernet interface.

Chapter 3 GP-IB Interface (Optional)

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.

SHIFT+Panel Key
When *SHIFT+panel key* appears in a procedural explanation, it means to press the shift key so that its indicator lights, and then to press the indicated panel key. A setup menu for the item written in purple above the key that you pressed appears 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.

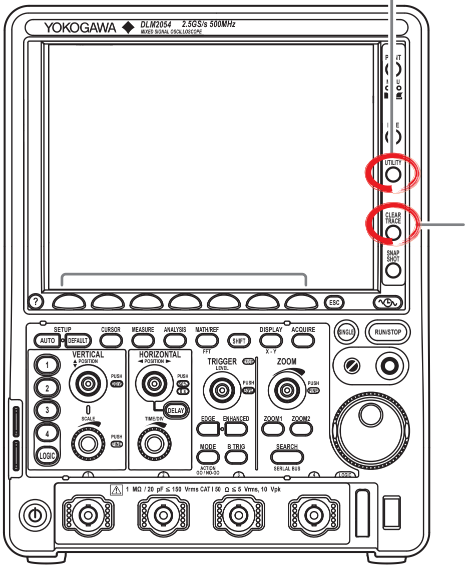
Symbol	Description	Example	Example
<>	A defined value	`CHANnel<x>` `<x>` = 1 to 4	`CHANNEL2`
{} |	Select an option in { } Exclusive OR	`COUPling {AC|DC|DC50|GND}`	`COUPLING AC`
[]	Can be omitted	`TRIGger [:SIMPle]:SLOPe`	`TRIGger:SLOPe`

Contents

1.1 Component Names and Functions

Front Panel

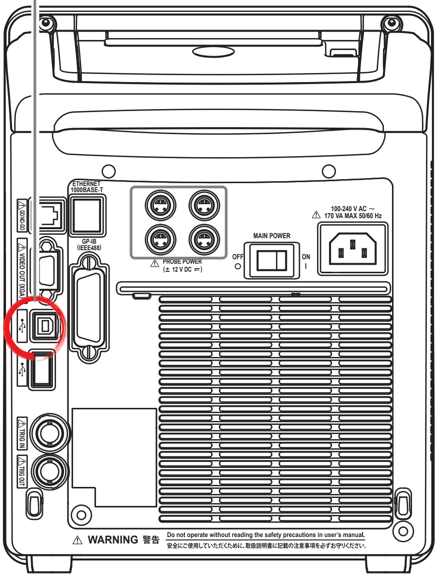
UTILITY key
Press this key to select the USB interface.



CLEAR TRACE key (SHIFT+CLEAR TRACE)
Press this key to switch from Remote mode to Local mode, which allows you to control the DLM2000 using front panel keys.
This key is disabled when Local Lockout (see page 1-2 for details) has been activated by a controller.

Rear Panel

USB port
This port is for connecting the DLM2000 to a controller (such as a PC) using a USB cable.


ETHERNET 1000BASE-T
GP-IB (IEEE488)
PROBE POWER (± 12 V DC)
MAIN POWER
OFF
ON
100-240 V AC 170 VA MAX 50/60 Hz
WARNING 警告
Do not operate without reading the safety precautions in user's manual.

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 DLM2000 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 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 (32bit, 64bit), Windows 8 (32bit, 64bit), Windows 8.1 (32bit, 64bit), or Windows 10 (32bit, 64bit) and a separate device driver

Data Transfer Rate

The following table contains approximations of how much time it takes for the DLM2000 to transmit waveform data.

Model:	710130
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	Byte Data	Word Data	ASCII Data
1250	Approx. 13 ms	Approx. 12 ms	Approx. 0.70 s
12500	Approx. 21 ms	Approx. 14 ms	Approx. 0.67 s
125000	Approx. 109 ms	Approx. 34 ms	Approx. 6.65 s
1250000	Approx. 981 ms	Approx. 221 ms	Approx. 66.45 s
12500000	Approx. 9690 ms	Approx. 2126 ms	Approx. 666.40 s
125000000	Approx. 96173 ms	Approx. 20730 ms	Approx. 6647.052 s

Switching between Remote and Local Modes

Switching from Local to Remote Mode

The DLM2000 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 DLM2000 is in Remote mode.
- All keys except **SHIFT+CLEAR TRACE** key are disabled.
- The Local mode settings are retained even when the DLM2000 switches to Remote mode.

Switching from Remote to Local Mode

When the DLM2000 is in Remote mode and you press **SHIFT+CLEAR TRACE** key, the DLM2000 switches to Local mode. However, this does not work if the DLM2000 has received a `:COMMunicate:LOCKout ON` command from the PC. The DLM2000 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 DLM2000 is in Local mode.
- All keys are enabled.
- The Remote mode settings are retained even when the DLM2000 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 DLM2000 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 DLM2000.

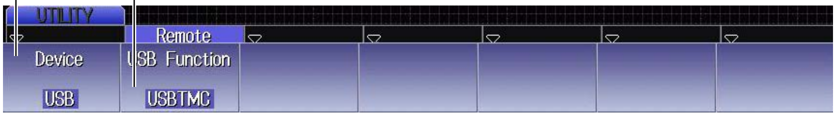
1.4 Configuring the DLM2000 USB Settings

This section explains the settings listed below. You must configure these settings when controlling the DLM2000 remotely through a USB interface.

- Communication interface

Opening the REMOTE CONTROL Menu

Press **UTILITY**, 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

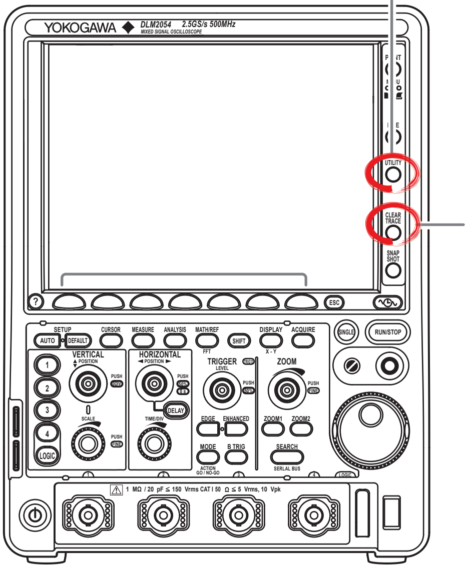
Note

- Only use the selected communication interface. If you send commands simultaneously from another communication interface that has not been selected, the DLM2000 will not execute the commands properly.
- To remotely control the DLM2000 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 DLM2000. Turn off the DLM2000 power switch, wait ten seconds or more, and then turn on the switch.
    - Install the YOKOGAWA USB driver (YKMUSB) on your PC. For information about how to obtain the YOKOGAWA USB driver (YKMUSB), contact your nearest YOKOGAWA dealer. You can also access the YOKOGAWA USB driver download webpage and download the driver.
      http://tmi.yokogawa.com/service-support/downloads/
- Do not use USB drivers (or software) supplied by other companies.

2.1 Component Names and Functions

Front Panel

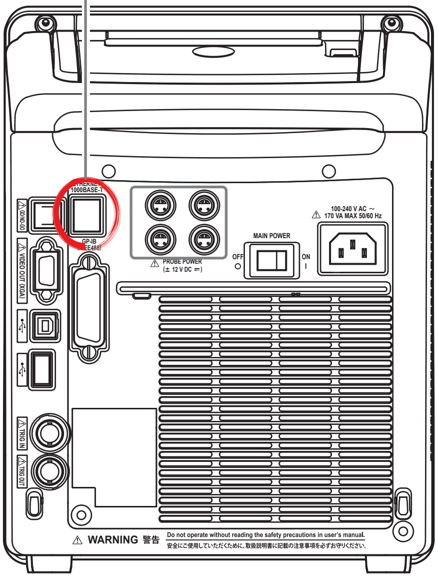
UTILITY key
Press this key to select the Ethernet interface.



CLEAR TRACE key (SHIFT+CLEAR TRACE)
Press this key to switch from Remote mode to Local mode, which allows you to control the DLM2000 using front panel keys.
This key is disabled when Local Lockout (see page 2-2 for details) has been activated by a controller.

Rear Panel

Ethernet port
This port is for connecting the DLM2000 to a controller (such as a PC) using an Ethernet cable.


ETHERNET 1000BASE-T
GP-IB (IEEE488)
PROBE POWER (± 12 V DC)
MAIN POWER
OFF
ON
100-240 V AC ~ 170 VA MAX 50/60 Hz
VIDEO OUT (XGA)
TRIG IN
TRIG OUT
WARNING 警告
Do not operate without reading the safety precautions in user's manual.
安全にご使用していただくために、取扱説明書に記載の注意事項を必ずお守りください。

2.2 Ethernet Interface Features and 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 DLM2000 can transmit measured and computed data.

The DLM2000 can transmit panel setting data and the status byte.

The DLM2000 can transmit 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 (32bit, 64bit), Windows 8 (32bit, 64bit), Windows 8.1 (32bit, 64bit), or Windows 10 (32bit, 64bit)

Data Transfer Rate

The following table contains approximations of how much time it takes for the DLM2000 to transmit waveform data.

Model:	710130
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	Byte Data	Word Data	ASCII Data
1250	Approx. 14 ms	Approx. 13 ms	Approx. 0.80 s
12500	Approx. 23 ms	Approx. 16 ms	Approx. 0.68 s
125000	Approx. 115 ms	Approx. 43 ms	Approx. 6.68 s
1250000	Approx. 1017 ms	Approx. 288 ms	Approx. 64.34 s
12500000	Approx. 10063 ms	Approx. 2760 ms	Approx. 659.26 s
125000000	Approx. 100055 ms	Approx. 27141 ms	Approx. 6676.55 s

Switching between Remote and Local Modes

Switching from Local to Remote Mode

The DLM2000 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 DLM2000 is in Remote mode.
- All keys except **SHIFT+CLEAR TRACE** key are disabled.
- The Local mode settings are retained even when the DLM2000 switches to Remote mode.

Switching from Remote to Local Mode

When the DLM2000 is in Remote mode and you press **SHIFT+CLEAR TRACE** key, the DLM2000 switches to Local mode. However, this does not work if the DLM2000 has received a `:COMMunicate:LOCKout ON` command from the PC. The DLM2000 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 DLM2000 is in Local mode.
- All keys are enabled.
- The Remote mode settings are retained even when the DLM2000 switches to Local mode.

Note

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

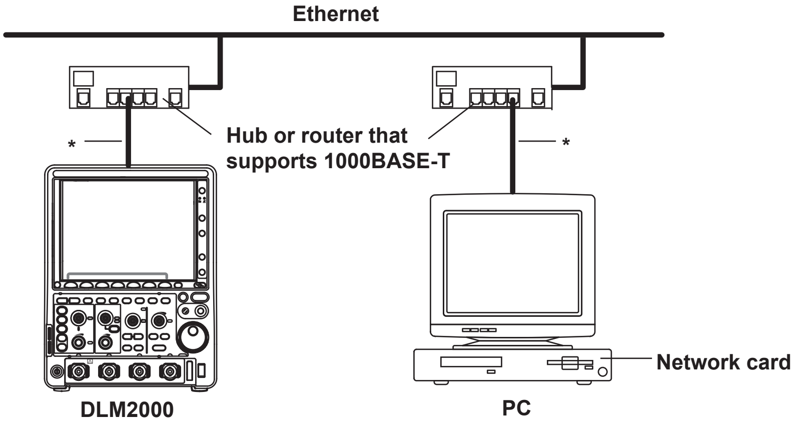
Setting the Timeout Value

If the DLM2000 is not accessed for a specified amount of time (set as a timeout value), the DLM2000 closes the connection to the network. You can set the timeout value in the range of 0 to 600 s. The default value is 0 s. For instructions on how to set the timer value, see section 2.4, "Network Configuration on the DLM2000."

2.3 Ethernet Interface Connection (Optional)

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 DLM2000 rear panel.



* Straight UTP or STP cable

Notes about Connections

- To connect the DLM2000 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 DLM2000 to a network, see section 18.1, “Connecting the DLM2000 to a Network” in the *DLM2000 Series User’s Manual (IM710105-02E)*.

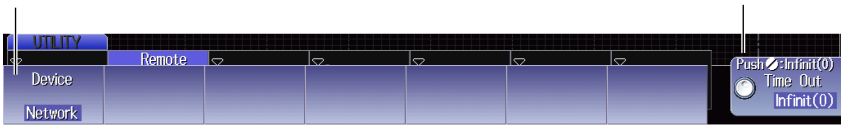
2.4 Configuring the DLM2000 Network Settings

This section explains the settings listed below. You must configure these settings when controlling the DLM2000 remotely through a ethernet interface.

- Communication interface
- User name and password

Opening the REMOTE CONTROL Menu

Press **UTILITY**, and then press the **REMOTE CONTROL** soft key to open the menu shown below.



Set the communication interface to Network

Set the timeout

Note

Only use the selected communication interface. If you send commands simultaneously from another communication interface that has not been selected, the DLM2000 will not execute the commands properly.

Configuring TCP/IP Settings

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

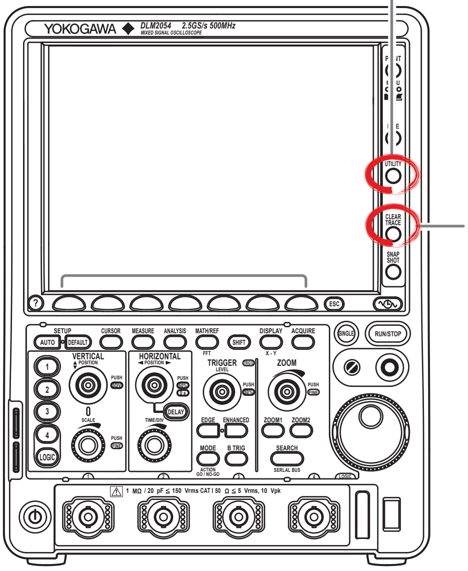
- IP address
- Subnet mask
- Default gateway

For instructions on how to specify these settings, see section 18.2, “Configuring TCP/IP Settings” in the *DLM2000 Series User’s Manual (IM710105-02E)*.

3.1 Component Names and Functions

Front Panel

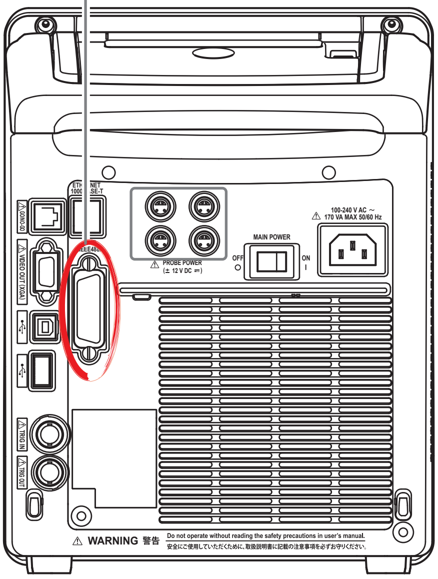
UTILITY key
Press this key to select the GP-IB interface.



CLEAR TRACE key (SHIFT+CLEAR TRACE)
Press this key to switch from Remote mode to Local mode, which allows you to control the DLM2000 using front panel keys.
This key is disabled when Local Lockout (see page 3-3 for details) has been activated by a controller.

Rear Panel

GP-IB port
This port is for connecting the DLM2000 to a controller (such as a PC) using a GP-IB cable.



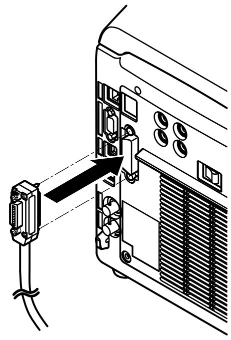
3.2 Connecting GP-IB Cables (Optional)

GP-IB Cable

The DLM2000 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.

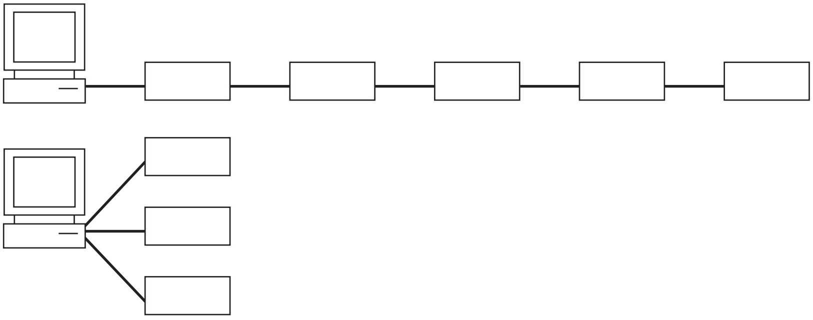
Connection Procedure

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 DLM2000 may not operate properly if the DLM2000 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 DLM2000 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 DLM2000 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 MLB2000 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 DLM2000 can transmit measured and computed data.
The DLM2000 can transmit panel setting data and the status byte.
The DLM2000 can transmit error codes when errors occur.

Note

Talk-only, listen-only, and controller capabilities are not available on the DLM2000.

Switching between Remote and Local Modes

Switching from Local to Remote Mode

When the DLM2000 is in Local mode and it receives a REN (Remote Enable) message from the PC, the DLM2000 switches to Remote mode.

- “REMOTE” appears at the top center of the screen once the DLM2000 is in Remote mode.
- All keys except **SHIFT+CLEAR TRACE** key are disabled.
- The settings in Local mode are retained even when the DLM2000 switches to Remote mode.

Switching from Remote to Local Mode

When the DLM2000 is in Remote mode and you press **SHIFT+CLEAR TRACE** key, the DLM2000 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 DLM2000 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**+**CLEAR TRACE** 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 DLM2000 to transmit waveform data.

Model: 710130

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	Byte Data	Word Data	ASCII Data
1250	Approx. 14 ms	Approx. 15 ms	Approx. 0.13 s
12500	Approx. 32 ms	Approx. 35 ms	Approx. 0.76 s
125000	Approx. 211 ms	Approx. 237 ms	Approx. 7.45 s
1250000	Approx. 2005 ms	Approx. 2267 ms	Approx. 74.31 s
12500000	Approx. 19982 ms	Approx. 22614 ms	Approx. 744.53 s
125000000	Approx. 199146 ms	Approx. 225424 ms	Approx. 7438.84 s

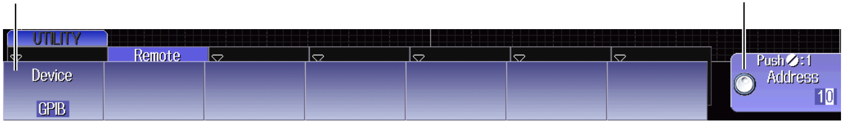
3.5 Configuring the DLM2000 GP-IB Settings

This section explains the settings listed below. You must configure these settings when controlling the DLM2000 remotely through a GP-IB interface.

- Communication interface
- GP-IB address

Opening the REMOTE CONTROL Menu

Press **UTILITY**, and then press the **REMOTE CONTROL** soft key to open the following menu.



Device
GPIB

Address
10

Set the communication interface to GP-IB

Set the address (0 to 30)

Note

- Only use the selected communication interface. If you send commands simultaneously from another communication interface that has not been selected, the DLM2000 will not execute the commands properly.
- When the controller is communicating with the DLM2000 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 DLM2000 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**+**CLEAR TRACE** 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 DLM2000 is designated as a listener or a talker. The following five commands are available.

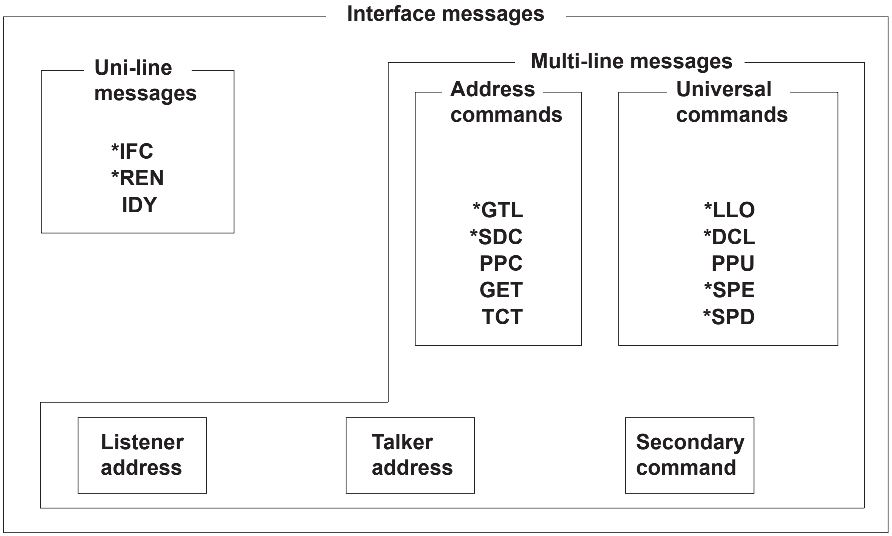
  Commands available to a device designated as a listener
  - GTL (Go To Local)
  - SDC (Selected Device Clear)
  - PPC (Parallel Poll Configure)
  - GET (Group Execute Trigger)

  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 DLM2000 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

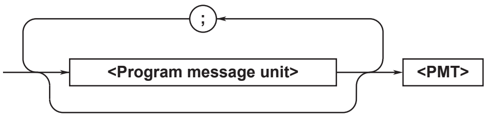
Messages

Messages are used to exchange information between the controller and the DLM2000. Messages that are sent from the controller to the DLM2000 are called program messages, and messages that are sent from the DLM2000 back to the controller are called response messages.

If a program message contains a command that requests a response (query), the DLM2000 returns a response message upon receiving the program message. The DLM2000 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 DLM2000 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

$\underbrace{\texttt{:ACQuire:MODE NORMal}}_{\text{Unit}}\texttt{;}\underbrace{\texttt{HRMode 1}}_{\text{Unit}}\texttt{<PMT>}$

<PMT>

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

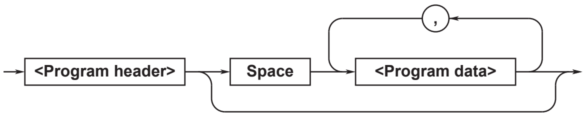
`NL` (new line): Same as LF (line feed). ASCII code “0AH”

`^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>

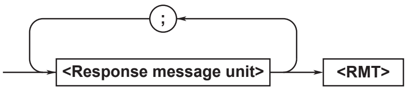
Attach program data if there are conditions that are required to execute a command. Separate the program data from the header with a space (ASCII code 20H). If there are multiple data values, separate each data value with a comma.

For details, see page 4-5.

Example $\underbrace{\texttt{:ACQuire:MODE}}_{\text{Header}}\ \underbrace{\texttt{NORMal}}_{\text{Data}}\texttt{<PMT>}$

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

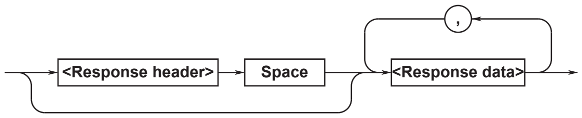
$\underbrace{\texttt{:ACQUIRE:MODE NORMAL}}_{\text{Unit}}\texttt{;}\underbrace{\texttt{HRMode 1}}_{\text{Unit}}\texttt{<RMT>}$

<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:



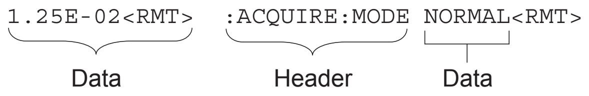
<Response Header>

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

<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



```
1.25E-02<RMT>  :ACQUIRE:MODE NORMAL<RMT>
```

If there are multiple queries in a program message, responses are returned in the same order that the queries were received in. The DLM2000 returns a single response message unit to most queries, but there are queries that the DLM2000 returns multiple units to. The first response message unit always corresponds to the first query, but the nth response unit may not necessarily correspond to the nth 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 DLM2000 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 DLM2000 may not necessary return responses.

Deadlock

The DLM2000 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 DLM2000 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.

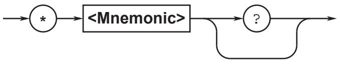
4.2 Commands

Command

There are three types of commands (program headers) that a controller may send to the DLM2000. The commands differ in their program header formats.

Common Command Header

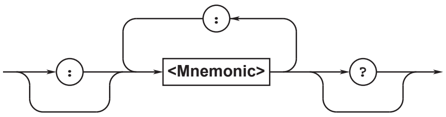
Commands that are defined in USBTMC-USB488 are called common commands. The header format of a common command is shown below. Be sure to include an asterisk (*) at the beginning of a common command.



Common command example `*CLS`

Compound Header

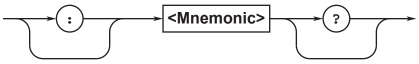
Other commands that are specific to the DLM2000 are classified and arranged in a hierarchy according to their functions. The compound header syntax is shown below. Be sure to use a colon to specify a lower hierarchical level.



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:AVERage:EWEight
  :ACQuire:REPetitive
  :ACQuire:RLENgth
  :ACQuire:INTERLeave
  ```

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

  The DLM2000 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;
  INTERLeave 1<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;
  INTERLeave 1<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:INTERLeave 1<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 DLM2000 to transmit all the lower level settings.

Example
```
:ACQUIRE? ->
:ACQUIRE:AVERAGE:COUNT
2;EWEIGHT 16;:ACQUIRE:HRMODE
0;INTERLEAVE 0;INTERPOLATE 1;MODE
NORMAL;REPETITIVE 0;RLENGTH 12500
```

The response to an upper-level query can be sent back to the DLM2000 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 DLM2000 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: `HISTory[:CURRent]:MODE` can be written as `HIST:MODE`.
  However, the last section enclosed in brackets cannot be omitted in an upper-level query.
  Example: `HISTory?` and `HISTory:CURRent?` are different queries.

4.3 Responses

Response

When the controller sends a query with a question mark, the DLM2000 returns a response message to the query. The DLM2000 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 DLM2000 will attach headers to.

  Example
  ```
  :MEASure:TRACe1:AREA1:PTOPeak:
  VALue?<PMT> -> 10.0E+00<RMT>
  ```

If You Want the DLM2000 to Return Responses without Headers

You can configure the DLM2000 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 DLM2000 normally returns response headers with the lower-case section removed. You can configure the DLM2000 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|GND}`)
<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 `-> MATH1:UNIT:USERdefine "VOLT"`)
<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 DLM2000 can receive decimal values that are sent from the controller in any form, from <NR1> to <NR3>. This is expressed as <NRf>.
- The DLM2000 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|GND}`	`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 DLM2000 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.

<Filename>

<Filename> is data that indicates a file name. The following types of expressions are possible.

Form	Example
{<NRf>|<character data>|<string data>}	`1 CASE "CASE"`

- <NRf> is rounded to an 8-digit integer and converted to ASCII code. The result is the file name (example: 1 becomes “`00000001`”). Negative values are not allowed.
- For <character data>, the first 12 characters become the file name.
- For <string data>, the first 259 characters become the file name.
- Response messages are always expressed in the <string data> form.
- For information about the number of characters in a file name expressed in the <string data form>, see the *DLM2000 User’s Manual*.

<Block Data>

<Block data> is any 8-bit data. It is only used in response messages on the DLM2000. 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 DLM2000 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 SINGle;: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
  SINGle;: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
  SINGle;: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
  SINGle<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.	5-56
:ACQuire:AVERage?	Queries all of the settings related to averaging and the number of waveform acquisitions.	5-56
:ACQuire:AVERage:COUNt	Sets or queries the attenuation constant for exponential averaging or the average count for linear averaging for Averaging mode.	5-56
:ACQuire:COUNt	Sets or queries the number of waveform acquisitions in Normal, Envelope, and Averaging modes.	5-56
:ACQuire:INTerleave	Sets or queries the on/off status of Interleave mode.	5-56
:ACQuire:MODE	Sets or queries the waveform acquisition mode.	5-56
:ACQuire:RESolution	Sets or queries the on/off status of High Resolution mode.	5-56
:ACQuire:RLENgth	Sets or queries the record length.	5-56
:ACQuire:SAMPling	Sets or queries the sampling mode.	5-56
ANALysis Group		
:ANALysis?	Queries all of the settings for the analysis feature.	5-57
:ANALysis:AHIStogram<x>?	Queries all of the settings for the waveform histogram feature.	5-57
:ANALysis:AHIStogram<x>:DISPlay	Sets or queries the on/off status of a waveform histogram display.	5-57
:ANALysis:AHIStogram<x>:HORizontal	Sets or queries the horizontal range of a waveform histogram.	5-57
:ANALysis:AHIStogram<x>:MEASure?	Queries all of the settings for waveform histogram measurement (including the mode on/off status).	5-57
:ANALysis:AHIStogram<x>:MEASure:MODE	Sets or queries the measurement mode of a waveform histogram.	5-57
:ANALysis:AHIStogram<x>:MEASure:PARameter?	Queries all of the measurement settings for a waveform histogram when the histogram parameter measurement mode is set to Param.	5-57
:ANALysis:AHIStogram<x>:MEASure:PARameter:ALL	Collectively turns on or off all of the measurement items of a waveform histogram.	5-57
:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>?	Queries all of the settings for the specified measurement item of a waveform histogram.	5-57
:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>:STATe	Sets or queries the on/off status of the specified measurement item of a waveform histogram.	5-58
:ANALysis:AHIStogram<x>:MEASure:PARameter:<Parameter>:VALue?	Queries the automatically measured value for the specified measurement item of a waveform histogram.	5-58
:ANALysis:AHIStogram<x>:MEASure:PARameter:HPOSition<y>	Sets or queries the horizontal position of Cursor1 or Cursor 2 for a waveform histogram.	5-58
:ANALysis:AHIStogram<x>:MEASure:PARameter:VPOSition<y>	Sets or queries the vertical position of Cursor1 or Cursor 2 for a waveform histogram.	5-58
:ANALysis:AHIStogram<x>:MODE	Sets or queries the target axis of a waveform histogram.	5-59
:ANALysis:AHIStogram<x>:RANGe	Sets or queries the measurement source window of a waveform histogram.	5-59
:ANALysis:AHIStogram<x>:TRACe	Sets or queries the source waveform of a waveform histogram.	5-59
:ANALysis:AHIStogram<x>:VERTical	Sets or queries the vertical range of a waveform histogram.	5-59
:ANALysis:PANalyze<x>?	Queries all power supply analysis settings.	5-59
:ANALysis:PANalyze<x>:I2T?	Queries all Joule integral settings.	5-59
:ANALysis:PANalyze<x>:I2T:MATH	Sets or queries the on/off status of the Joule integral waveform display.	5-59
:ANALysis:PANalyze<x>:I2T:MEASure?	Queries all Joule integral automatic measurement settings.	5-59
:ANALysis:PANalyze<x>:I2T:MEASure:I2T?	Queries Joule integral settings.	5-59
:ANALysis:PANalyze<x>:I2T:MEASure:I2T:COUNt?	Queries the continuous statistical processing count of a Joule integral.	5-59
:ANALysis:PANalyze<x>:I2T:MEASure:I2T:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a Joule integral.	5-60
:ANALysis:PANalyze<x>:I2T:MEASure:I2T:STATe	Sets or queries whether or not a Joule integral will be measured.	5-60

5.1 List of Commands

Command	Function	Page
:ANALysis:PANalyze<x>:I2T:MEASure:I2T:VALue?	Queries an automatically measured value of a Joule integral.	5-60
:ANALysis:PANalyze<x>:I2T:RANGe	Sets or queries the measurement source window.	5-60
:ANALysis:PANalyze<x>:I2T:SCALe?	Queries all scaling settings.	5-60
:ANALysis:PANalyze<x>:I2T:SCALe:CENTer	Sets or queries the center value for manual scaling.	5-61
:ANALysis:PANalyze<x>:I2T:SCALe:MODE	Sets or queries the scaling mode.	5-61
:ANALysis:PANalyze<x>:I2T:SCALe:SENSitivity	Sets or queries the span from the center value for manual scaling.	5-61
:ANALysis:PANalyze<x>:I2T:TRANge (Time Range)	Sets or queries the measurement time period.	5-61
:ANALysis:PANalyze<x>:HARMonics?	Queries all harmonic analysis settings.	5-61
:ANALysis:PANalyze<x>:HARMonics:CCLass?	Queries all class C harmonic analysis settings.	5-61
:ANALysis:PANalyze<x>:HARMonics:CCLass:GETLambda	Queries the current power factor for class C harmonic analysis.	5-61
:ANALysis:PANalyze<x>:HARMonics:CCLass:LAMBda	Sets or queries the power factor for class C harmonic analysis.	5-61
:ANALysis:PANalyze<x>:HARMonics:CCLass:MAXCurrent	Sets or queries the fundamental current value for class C harmonic analysis.	5-62
:ANALysis:PANalyze<x>:HARMonics:CCLass:OPOWer	Sets or queries whether or not the active power for class C harmonic analysis exceeds 25 W.	5-62
:ANALysis:PANalyze<x>:HARMonics:CLASs	Sets or queries the class of the device under harmonic analysis.	5-62
:ANALysis:PANalyze<x>:HARMonics:DCLass?	Queries all class D harmonic analysis settings.	5-62
:ANALysis:PANalyze<x>:HARMonics:DCLass:POWer	Sets or queries the power value for class D harmonic analysis.	5-62
:ANALysis:PANalyze<x>:HARMonics:DMODe	Sets or queries the harmonic analysis display mode.	5-62
:ANALysis:PANalyze<x>:HARMonics:DETail?	Queries all harmonic analysis result list settings.	5-62
:ANALysis:PANalyze<x>:HARMonics:DETail:DISPlay	Sets or queries the harmonic analysis result list display mode.	5-62
:ANALysis:PANalyze<x>:HARMonics:DETail:LIST:ITEM?	Queries the items displayed in the harmonic analysis result list.	5-63
:ANALysis:PANalyze<x>:HARMonics:DETail:LIST:VALue?	Queries all of the data for the specified analysis number in the harmonic analysis result list.	5-63
:ANALysis:PANalyze<x>:HARMonics:GROuping	Sets or queries the harmonic analysis grouping.	5-63
:ANALysis:PANalyze<x>:HARMonics:SPOint	Sets or queries the harmonic analysis computation start point.	5-63
:ANALysis:PANalyze<x>:HARMonics:SVOLtage	Sets or queries the power supply voltage for harmonic analysis.	5-63
:ANALysis:PANalyze<x>:SETup?	Queries all power supply analysis input settings.	5-63
:ANALysis:PANalyze<x>:SETup:ADESkew	Executes automatic deskewing for power supply analysis.	5-63
:ANALysis:PANalyze<x>:SETup:I?	Queries all current input channel settings for power supply analysis.	5-64
:ANALysis:PANalyze<x>:SETup:I:DESKew	Sets or queries the deskew setting of the current input channel for power supply analysis.	5-64
:ANALysis:PANalyze<x>:SETup:I:INPut	Sets or queries the current input channel for power supply analysis.	5-64
:ANALysis:PANalyze<x>:SETup:I:PROBe	Queries all current-to-voltage conversion ratio settings for the current input channel probe for power supply analysis.	5-64
:ANALysis:PANalyze<x>:SETup:U?	Queries all voltage input channel settings for power supply analysis.	5-64
:ANALysis:PANalyze<x>:SETup:U:DESKew	Sets or queries the deskew setting of the voltage input channel for power supply analysis.	5-64
:ANALysis:PANalyze<x>:SETup:U:INPut	Sets or queries the voltage input channel for power supply analysis.	5-64
:ANALysis:PANalyze<x>:SETup:U:PROBe	Sets or queries the probe attenuation setting of the voltage input channel for power supply analysis.	5-64

Command	Function	Page
:ANALysis:PANalyze<x>:SETup:RTRace	Sets or queries the reference trace for power supply analysis deskewing.	5-65
:ANALysis:PANalyze<x>:SOA?	Queries all XY display (safe operating area) settings.	5-65
:ANALysis:PANalyze<x>:SOA:CURSor?	Queries all XY display (safe operating area) cursor measurement settings.	5-65
:ANALysis:PANalyze<x>:SOA:CURSor:X<y>?	Queries all XY display (safe operating area) horizontal cursor settings.	5-65
:ANALysis:PANalyze<x>:SOA:CURSor:X<y>:POSition	Sets or queries a horizontal cursor position in the XY display (safe operating area).	5-65
:ANALysis:PANalyze<x>:SOA:CURSor:X<y>:VALue?	Sets or queries the voltage value of a horizontal cursor in the XY display (safe operating area).	5-65
:ANALysis:PANalyze<x>:SOA:CURSor:Y<y>?	Queries all XY display (safe operating area) vertical cursor settings.	5-65
:ANALysis:PANalyze<x>:SOA:CURSor:Y<y>:POSition	Sets or queries a vertical cursor position in the XY display (safe operating area).	5-65
:ANALysis:PANalyze<x>:SOA:CURSor:Y<y>:VALue?	Sets or queries the voltage value of a vertical cursor in the XY display (safe operating area).	5-65
:ANALysis:PANalyze<x>:SOA:MODE	Sets or queries the automated measurement mode of the XY display (safe operating area).	5-66
:ANALysis:PANalyze<x>:SOA:TRANge (Time Range)	Sets or queries the range of the T-Y waveform to display in the XY display (safe operating area).	5-66
:ANALysis:PANalyze<x>:SOA:VTDisplay	Sets or queries whether or not to display the VT waveform in the XY display (safe operating area).	5-66
:ANALysis:PANalyze<x>:SWLoss?	Queries all switching loss settings.	5-66
:ANALysis:PANalyze<x>:SWLoss:CYCLe	Sets or queries the on/off status of cycle mode.	5-66
:ANALysis:PANalyze<x>:SWLoss:DPROximal?	Queries all distal, mesial, and proximal settings.	5-66
:ANALysis:PANalyze<x>:SWLoss:DPROximal:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-66
:ANALysis:PANalyze<x>:SWLoss:DPROximal:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-67
:ANALysis:PANalyze<x>:SWLoss:DPROximal:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-67
:ANALysis:PANalyze<x>:SWLoss:DTYPe	Sets or queries the device type for total loss computation.	5-67
:ANALysis:PANalyze<x>:SWLoss:ILEVel	Sets or queries the current level used to determine the zero loss period for total loss computation.	5-67
:ANALysis:PANalyze<x>:SWLoss:MATH	Sets or queries the on/off status of the power waveform display.	5-67
:ANALysis:PANalyze<x>:SWLoss:MEASure?	Queries all of the settings for the automated measurement of power supply analysis parameters.	5-67
:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>?	Queries the setting of a power supply analysis parameter.	5-67
:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:COUNt?	Queries the continuous statistical processing count of a power supply analysis parameter.	5-68
:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a power supply analysis parameter.	5-68
:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:STATe	Sets or queries the on/off status of a power supply analysis parameter.	5-68
:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:VALue?	Queries an automatically measured value of a power supply analysis parameter.	5-68
:ANALysis:PANalyze<x>:SWLoss:METHod	Sets or queries the calculation method for high and low points.	5-68
:ANALysis:PANalyze<x>:SWLoss:RANGe	Sets or queries the measurement source window.	5-69
:ANALysis:PANalyze<x>:SWLoss:RDS	Sets or queries the on-resistance value for total loss computation.	5-69
:ANALysis:PANalyze<x>:SWLoss:SCALe?	Queries all scaling settings.	5-69
:ANALysis:PANalyze<x>:SWLoss:SCALe:CENTer	Sets or queries the center value for manual scaling.	5-69

5.1 List of Commands

Command	Function	Page
:ANALysis:PANalyze<x>:SWLoss:SCALe:MODE	Sets or queries the scaling mode.	5-69
:ANALysis:PANalyze<x>:SWLoss:SCALe:SENSitivity	Sets or queries the span from the center value for manual scaling.	5-69
:ANALysis:PANalyze<x>:SWLoss:TRANge (Time Range)	Sets or queries the measurement time period.	5-69
:ANALysis:PANalyze<x>:SWLoss:ULEVel	Sets or queries the voltage level used to determine the loss calculation period for total loss computation.	5-70
:ANALysis:PANalyze<x>:SWLoss:UNIT	Sets or queries the power unit.	5-70
:ANALysis:PANalyze<x>:SWLoss:VCE	Sets or queries the collector-emitter saturation voltage value for total loss computation.	5-70
:ANALysis:PANalyze<x>:TYPE	Sets or queries the type of power supply analysis.	5-70
:ANALysis:PMEAsure<x>?	Queries all power measurement settings.	5-70
:ANALysis:PMEAsure<x>:IDPRoximal?	Queries all distal, mesial, and proximal settings.	5-70
:ANALysis:PMEAsure<x>:IDPRoximal:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-70
:ANALysis:PMEAsure<x>:IDPRoximal:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-70
:ANALysis:PMEAsure<x>:IDPRoximal:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-71
:ANALysis:PMEAsure<x>:IMEThod	Sets or queries the high and low points.	5-71
:ANALysis:PMEAsure<x>:INDicator	Sets or queries the measurement location indicator.	5-71
:ANALysis:PMEAsure<x>:MEASure?	Queries all the settings for automated measurement of power measurement parameters.	5-71
:ANALysis:PMEAsure<x>:MEASure:<Parameter>?	Queries the setting of a power measurement parameter.	5-71
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:COUNt?	Queries the normal statistical processing count of the power measurement parameter.	5-71
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a power measurement parameter.	5-71
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:STATe	Queries the on/off status of the power measurement parameter.	5-72
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:VALue?	Queries an automatically measured value of a power measurement parameter.	5-72
:ANALysis:PMEAsure<x>:MEASure:ALL	Collectively turns on or off the power measurement parameter.	5-72
:ANALysis:PMEAsure<x>:MODE	Sets or queries the on/off status of power measurement.	5-72
:ANALysis:PMEAsure<x>:RANGe	Sets or queries the measurement source window.	5-72
:ANALysis:PMEAsure<x>:SETup?	Queries all power measurement input settings.	5-72
:ANALysis:PMEAsure<x>:SETup:ADESkew	Executes auto deskewing for power measurement.	5-72
:ANALysis:PMEAsure<x>:SETup:I?	Queries all current input channel settings for power measurement.	5-72
:ANALysis:PMEAsure<x>:SETup:I:DESKew	Set or queries auto deskewing of the current input channel for power measurement.	5-72
:ANALysis:PMEAsure<x>:SETup:I:PROBe	Queries all current-to-voltage conversion ratio settings of the probe connected to the current input channel for power measurement.	5-73
:ANALysis:PMEAsure<x>:SETup:RTRace	Sets or queries the deskewing source trace for power measurement.	5-73
:ANALysis:PMEAsure<x>:SETup:U?	Queries all voltage input channel settings for power measurement.	5-73
:ANALysis:PMEAsure<x>:SETup:U:DESKew	Set or queries auto deskewing of the voltage input channel for power measurement.	5-73
:ANALysis:PMEAsure<x>:SETup:U:PROBe	Queries all voltage-to-voltage conversion ratio settings of the probe connected to the voltage input channel for power measurement.	5-73
:ANALysis:PMEAsure<x>:TRANge	Sets or queries the measurement time period.	5-73
:ANALysis:PMEAsure<x>:UDPRoximal?	Queries all distal, mesial, and proximal settings.	5-73
:ANALysis:PMEAsure<x>:UDPRoximal:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-73
:ANALysis:PMEAsure<x>:UDPRoximal:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-74

Command	Function	Page
:ANALysis:PMEAsure<x>:UDPRoximal:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-74
:ANALysis:PMEAsure<x>:UMEThod	Sets or queries the high and low points.	5-74
:ANALysis:PMEAsure<x>:UNIT	Sets or queries the watthour unit.	5-74
:ANALysis:PMEAsure<x>:USER<y>?	Queries all automatic measurement settings for a Calc item.	5-74
:ANALysis:PMEAsure<x>:USER<y>:COUNt?	Queries the statistical processing count of the automatically measured value of a Calc item.	5-74
:ANALysis:PMEAsure<x>:USER<y>:DEFine	Sets or queries the expression for the automatically measured value of a Calc item.	5-74
:ANALysis:PMEAsure<x>:USER<y>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value that is calculated on the automatically measured value of a Calc item.	5-74
:ANALysis:PMEAsure<x>:USER<y>:NAME	Sets or queries the name of a Calc item.	5-74
:ANALysis:PMEAsure<x>:USER<y>:STATe	Sets or queries the on/off status of automated measurement of a Calc item.	5-75
:ANALysis:PMEAsure<x>:USER<y>:UNIT	Sets or queries the unit of a Calc item.	5-75
:ANALysis:PMEAsure<x>:USER<y>:VALue?	Queries the automatically measured value of a Calc item.	5-75
:ANALysis:WAIT?	Waits for the completion of automated measurement with a set timeout.	5-75

ASETup Group

:ASETup:EXECute	Executes auto setup.	5-76
:ASETup:UNDO	Undoes auto setup.	5-76

CALibrate Group

:CALibrate?	Queries all calibration settings.	5-77
:CALibrate[:EXECute]	Executes calibration.	5-77
:CALibrate:MODE	Sets or queries the on/off status of auto calibration.	5-77

CHANnel Group

:CHANnel<x>?	Queries all vertical axis settings of a channel.	5-78
:CHANnel<x>:ASCale[:EXECute]	Executes auto scaling of a channel.	5-78
:CHANnel<x>:BWIDth	Sets or queries the input filter setting of a channel.	5-78
:CHANnel<x>:COUPling	Sets or queries the input coupling setting of a channel.	5-78
:CHANnel<x>:DESKew	Sets or queries the deskew setting of a channel.	5-78
:CHANnel<x>:DISPlay	Sets or queries the display on/off status of a channel.	5-78
:CHANnel<x>:INVert	Sets or queries the on/off status of invert mode, which inverts the waveform display.	5-78
:CHANnel<x>:LABel?	Queries all waveform label settings of a channel.	5-78
:CHANnel<x>:LABel[:DEFine]	Sets or queries the waveform label of a channel.	5-78
:CHANnel<x>:LABel:DISPlay	Sets or queries the waveform label display on/off status of a channel.	5-79
:CHANnel<x>:LSCale?	Queries all linear scaling settings of a channel.	5-79
:CHANnel<x>:LSCale:AVALue	Sets or queries a scaling coefficient A value.	5-79
:CHANnel<x>:LSCale:BVALue	Sets or queries an offset B value.	5-79
:CHANnel<x>:LSCale:MODE	Sets or queries the on/off status of linear scaling.	5-79
:CHANnel<x>:LSCale:UNIT	Sets or queries a unit that is attached to linear scaling results.	5-79
:CHANnel<x>:OFFSet	Sets or queries the offset voltage of a channel.	5-79
:CHANnel<x>:POSition	Sets or queries the vertical position of a channel.	5-79
:CHANnel<x>:PROBe?	Queries all probe attenuation settings of a channel.	5-79
:CHANnel<x>:PROBe:DZCalibrate	Performs demagnetization and zero adjustment on each channel.	5-79
:CHANnel<x>:PROBe[:MODE]	Sets or queries the probe attenuation setting of a channel.	5-80
:CHANnel<x>:PROBe:PZCalibrate	Executes zero calibration of current probes for a channel.	5-80
:CHANnel<x>:VARiable	Sets or queries the voltage scale of a channel in 0.01 V/division steps.	5-80
:CHANnel<x>:VDIV	Sets or queries the voltage scale (in V/division) of a channel.	5-80

CLEar Group

:CLEar	Clears traces.	5-81

Command	Function	Page
COMMunicate Group		
:COMMunicate?	Queries all communication settings.	5-82
:COMMunicate:HEADer	Sets or queries whether or not a header is added to the response to a query. (Example with header:CHANNEL1:PROBE:MODE 10. Example without header:10.)	5-82
:COMMunicate:LOCKout	Sets or clears local lockout.	5-82
:COMMunicate:OPSE(Operation Pending Status Enable register)	Sets or queries the overlap command that is used by the *OPC, *OPC?, and *WAI commands.	5-82
:COMMunicate:OPSR? (Operation Pending Status Register)	Queries the operation pending status register.	5-82
:COMMunicate:OVERlap	Sets or queries the commands that operate as overlap commands.	5-82
:COMMunicate:REMote	Sets remote or local. On is remote mode.	5-82
:COMMunicate:VERBose	Sets or queries whether the response to a query is returned fully spelled out (example:CHANNEL1:PROBE:MODE 10) or using abbreviation (example:CHAN:PROB 10).	5-83
:COMMunicate:WAIT	Waits for a specified extended event to occur.	5-83
:COMMunicate:WAIT?	Creates the response that is returned when a specified extended event occurs.	5-83
CURSor Group		
:CURSor?	Queries all cursor measurement settings.	5-84
:CURSor[:TY]?	Queries all cursor settings.	5-84
:CURSor[:TY]:DEGRee?	Queries all angle cursor settings.	5-84
:CURSor[:TY]:DEGRee:ALL	Collectively turns on or off the measured values of the angle cursor on the T-Y display.	5-84
:CURSor[:TY]:DEGRee:D<x>?	Queries all angle cursor angle settings.	5-84
:CURSor[:TY]:DEGRee:D<x>:STATe	Sets or queries the on/off status of an angle cursor angle.	5-84
:CURSor[:TY]:DEGRee:D<x>:VALue?	Queries an angle cursor angle.	5-84
:CURSor[:TY]:DEGRee:DD?	Queries all of the settings for the angle difference between the angle cursors.	5-84
:CURSor[:TY]:DEGRee:DD:STATe	Sets or queries the on/off status of the angle difference value ΔD, which is measured between the angle cursors.	5-84
:CURSor[:TY]:DEGRee:DD:VALue?	Queries the angle difference value ΔD, which is measured between the angle cursors.	5-84
:CURSor[:TY]:DEGRee:DV?	Queries all of the settings for the ΔV value, which is measured between the angle cursors.	5-84
:CURSor[:TY]:DEGRee:DV:STATe	Sets or queries the on/off status of the ΔV value, which is measured between the angle cursors.	5-84
:CURSor[:TY]:DEGRee:DV:VALue?	Queries the ΔV value, which is measured between the angle cursors.	5-84
:CURSor[:TY]:DEGRee:JUMP	Makes an angle cursor in the T-Y display jump to the specified zoom waveform.	5-84
:CURSor[:TY]:DEGRee:POSition<x>	Sets or queries the position of an angle cursor.	5-85
:CURSor[:TY]:DEGRee:REFerence<x>	Sets or queries the position of the angle reference start point (Reference1) or the angle reference end point (Reference2).	5-85
:CURSor[:TY]:DEGRee:RVALue	Sets or queries the reference angle.	5-85
:CURSor[:TY]:DEGRee:TRACe	Sets or queries the source waveform that you want to measure using the angle cursors.	5-85
:CURSor[:TY]:DEGRee:UNIT	Sets or queries the unit of angle cursor measurement.	5-85
:CURSor[:TY]:DEGRee:V<x>?	Queries all angle cursor voltage settings.	5-85
:CURSor[:TY]:DEGRee:V<x>:STATe	Sets or queries the on/off status of an angle cursor voltage.	5-85
:CURSor[:TY]:DEGRee:V<x>:VALue?	Queries an angle cursor voltage.	5-85
:CURSor[:TY]:HORizontal?	Queries all ΔV cursor settings.	5-85
:CURSor[:TY]:HORizontal:ALL	Collectively turns on or off the measured values of the ΔV cursor on the T-Y display.	5-85
:CURSor[:TY]:HORizontal:DV?	Queries all of the settings for the vertical values between the ΔV cursors.	5-85
:CURSor[:TY]:HORizontal:DV:STATe	Sets or queries the on/off status of the vertical value measured between the ΔV cursors.	5-86
:CURSor[:TY]:HORizontal:DV:VALue?	Queries the vertical value between the ΔV cursors.	5-86
:CURSor[:TY]:HORizontal:POSition<x>	Sets or queries the position of a vertical cursor.	5-86
:CURSor[:TY]:HORizontal:TRACe	Sets or queries the source waveform that you want to measure using the ΔV cursors.	5-86

Command	Function	Page
:CURSor[:TY]:HORizontal:V<x>?	Queries all vertical axis settings for a ΔV cursor.	5-86
:CURSor[:TY]:HORizontal:V<x>:STATe	Sets or queries the on/off status of the vertical value measured for a ΔV cursor.	5-86
:CURSor[:TY]:HORizontal:V<x>:VALue?	Queries the vertical value of a ΔV cursor.	5-86
:CURSor[:TY]:MARKer?	Queries all marker cursor settings.	5-86
:CURSor[:TY]:MARKer:FORM	Sets or queries the marker cursor form.	5-86
:CURSor[:TY]:MARKer:M<x>?	Queries all settings for the specified marker.	5-86
:CURSor[:TY]:MARKer:M<x>:ALL	Collectively turns on or off the measured values of the marker cursor on the T-Y display.	5-86
:CURSor[:TY]:MARKer:M<x>:DT<x>?	Queries all of the settings related to time measurement between marker cursors.	5-86
:CURSor[:TY]:MARKer:M<x>:DT<x>:STATe	Sets or queries the on/off status of the time value measured between marker cursors.	5-87
:CURSor[:TY]:MARKer:M<x>:DT<x>:VALue?	Queries the time value between marker cursors.	5-87
:CURSor[:TY]:MARKer:M<x>:DV<x>?	Queries all of the settings related to vertical-axis measurement between marker cursors.	5-87
:CURSor[:TY]:MARKer:M<x>:DV<x>:STATe	Sets or queries the on/off status of the vertical value measured between marker cursors.	5-87
:CURSor[:TY]:MARKer:M<x>:DV<y>:VALue?	Queries the vertical value between marker cursors.	5-87
:CURSor[:TY]:MARKer:M<x>:JUMP	Makes a marker cursor in the T-Y display jump to the specified zoom waveform.	5-87
:CURSor[:TY]:MARKer:M<x>:POSition	Sets or queries the time value of a marker cursor.	5-87
:CURSor[:TY]:MARKer:M<x>:T?	Queries all time axis settings for a marker cursor.	5-87
:CURSor[:TY]:MARKer:M<x>:T:STATe	Sets or queries the on/off status of the time value of a marker cursor.	5-87
:CURSor[:TY]:MARKer:M<x>:T:VALue?	Queries the time value of a marker cursor.	5-87
:CURSor[:TY]:MARKer:M<x>:TRACe	Sets or queries the source waveform that you want to measure using the marker cursors.	5-88
:CURSor[:TY]:MARKer:M<x>:V?	Queries all vertical axis settings for a marker cursor.	5-88
:CURSor[:TY]:MARKer:M<x>:V:STATe	Sets or queries the on/off status of the vertical value of a marker cursor.	5-88
:CURSor[:TY]:MARKer:M<x>:V:VALue?	Queries the vertical value of a marker cursor.	5-88
:CURSor[:TY]:MARKer:FORM	Sets or queries the marker cursor form.	5-88
:CURSor[:TY]:TYPE	Sets or queries the cursor type.	5-88
:CURSor[:TY]:VERTical?	Queries all ΔT cursor settings.	5-88
:CURSor[:TY]:VERTical:ALL	Collectively turns on or off the measured values of the ΔT cursor on the T-Y display.	5-88
:CURSor[:TY]:VERTical:DT?	Queries all time axis settings for the ΔT cursors.	5-88
:CURSor[:TY]:VERTical:DT:STATe	Sets or queries the on/off status of the time value measured between the ΔT cursors.	5-88
:CURSor[:TY]:VERTical:DT:VALue?	Queries the time value between the ΔT cursors.	5-88
:CURSor[:TY]:VERTical:DV?	Queries all vertical axis settings for the ΔT cursors.	5-88
:CURSor[:TY]:VERTical:DV:STATe	Sets or queries the on/off status of the vertical value measured between the ΔT cursors.	5-88
:CURSor[:TY]:VERTical:DV:VALue?	Queries the vertical value between the ΔT cursors.	5-89
:CURSor[:TY]:VERTical:JUMP	Makes a vertical cursor in the T-Y display jump to the specified zoom waveform.	5-89
:CURSor[:TY]:VERTical:PERDt?	Queries all time axis settings for 1/ΔT, which is measured between the ΔT cursors.	5-89
:CURSor[:TY]:VERTical:PERDt:STATe	Sets or queries the on/off status of the 1/ΔT, which is measured between the ΔT cursors.	5-89
:CURSor[:TY]:VERTical:PERDt:VALue?	Queries the 1/ΔT value, which is measured between the ΔT cursors.	5-89
:CURSor[:TY]:VERTical:POSition<x>	Sets or queries the position of a ΔT cursor.	5-89
:CURSor[:TY]:VERTical:T<x>?	Queries all time axis settings for a ΔT cursor.	5-89
:CURSor[:TY]:VERTical:T<x>:STATe	Sets or queries the on/off status of the time value of a ΔT cursor.	5-89

Command	Function	Page
:CURSor[:TY]:VERTical:T<x>:VALue?	Queries the time value of a ΔT cursor.	5-89
:CURSor[:TY]:VERTical:TRACe	Sets or queries the source waveform that you want to measure using the ΔT cursors.	5-89
:CURSor[:TY]:VERTical:V<x>?	Queries all vertical axis settings for a ΔT cursor.	5-89
:CURSor[:TY]:VERTical:V<x>:STATe	Sets or queries the on/off status of the vertical value of a ΔT cursor.	5-89
:CURSor[:TY]:VERTical:V<x>:VALue?	Queries the vertical value of a ΔT cursor.	5-90

DISPlay Group

:DISPlay?	Queries all display settings.	5-91
:DISPlay:ACCumulate?	Queries all accumulated waveform display settings.	5-91
:DISPlay:ACCumulate:MODE	Sets or queries the accumulation mode.	5-91
:DISPlay:ACCumulate:PERSistence	Sets or queries the accumulation time.	5-91
:DISPlay:COLor?	Queries all waveform color settings.	5-91
:DISPlay:COLor:{CHANnel<x>|MATH<x>}	Sets or queries a waveform color.	5-91
:DISPlay:COLor:LSTate	Sets or queries the color of the logic-waveform state display.	5-91
:DISPlay:COLor:SERialbus<x>?	Queries the color settings of all trends of the specified serial bus.	5-91
:DISPlay:COLor:SERialbus<x>:TRENd<y>	Sets or queries the serial bus trend color.	5-91
:DISPlay:FORMat	Sets or queries the display format.	5-91
:DISPlay:FGRid	Sets or queries the on/off status of the fine grid display.	5-92
:DISPlay:GRATicule	Sets or queries the graticule (grid).	5-92
:DISPlay:INTENsity?	Queries all intensity settings for the display items.	5-92
:DISPlay:INTENsity[:WAVeform]	Sets or queries the waveform intensity.	5-92
:DISPlay:INTENsity:{CURSor|GRID|MARKer|ZBOX}	Sets or queries the intensity of a display item.	5-92
:DISPlay:INTERpolate	Sets or queries the interpolation method.	5-92
:DISPlay:MAPPing	Sets or queries the split screen waveform mapping mode.	5-92
:DISPlay:SMAPping?	Queries all waveform mappings to the split screen.	5-92
:DISPlay:SMAPping:{CHANnel<x>|MATH<x>} (Set Mapping)	Sets or queries the split screen mapping of a waveform.	5-92
:DISPlay:SVALue (Scale VALUE)	Sets or queries the on/off status of the scale value display.	5-92

FFT Group

:FFT<x>?	Queries all of the settings for the FFT analysis feature.	5-93
:FFT<x>:AVERage?	Queries all FFT analysis averaging settings.	5-93
:FFT<x>:AVERage:EWEight	Sets or queries the attenuation constant of exponential averaging in FFT analysis.	5-93
:FFT<x>:DATA?	Queries all the information related to the FFT waveform data that will be sent.	5-93
:FFT<x>:DATA:BYTeorder	Sets or queries the transmission byte order when the FFT waveform data format is binary.	5-93
:FFT<x>:DATA:END	Sets or queries the end point of the FFT waveform data that will be sent.	5-93
:FFT<x>:DATA:FORMat	Sets or queries the format of the FFT waveform data that will be sent.	5-93
:FFT<x>:DATA:LENGth?	Queries the total number of data points of the FFT waveform that will be sent.	5-93
:FFT<x>:DATA:SEND?	Queries the FFT waveform data.	5-93
:FFT<x>:DATA:STARt	Sets or queries the start point of the FFT waveform data that will be sent.	5-94
:FFT<x>:DISPlay	Sets or queries whether or not FFT analysis will be performed.	5-94
:FFT<x>:HORizontal?	Queries all horizontal axis settings for FFT analysis.	5-94
:FFT<x>:HORizontal:CSPan?	Queries all center and span settings for the horizontal axis for FFT analysis.	5-94
:FFT<x>:HORizontal:CSPan:CENTer	Sets or queries the center value of the horizontal axis for FFT analysis.	5-94
:FFT<x>:HORizontal:CSPan:SPAN	Sets or queries the span value of the horizontal axis for FFT analysis.	5-94
:FFT<x>:HORizontal:LRIGht?	Queries all of the setting for the left and right ends of the horizontal axis for FFT analysis.	5-94
:FFT<x>:HORizontal:LRIGht:RANGe	Sets or queries the range for the left and right ends of the horizontal axis for FFT analysis.	5-94
:FFT<x>:HORizontal:MODE	Sets or queries the FFT analysis horizontal axis mode.	5-94
:FFT<x>:LENGth	Sets or queries the number of FFT points for FFT analysis.	5-94

Command	Function	Page
:FFT<x>:MEASure?	Queries all automatic measurement settings for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer?	Queries all marker cursor measurement settings for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]?	Queries all Basic marker cursor item settings for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:ALL	Collectively turns on or off all Basic marker cursor items for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:DFRequency?	Queries all of the settings for frequency measurement between marker cursors for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:DFRequency:STATe	Sets or queries the on/off status of the frequency measured between marker cursors for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:DFRequency:VALue?	Queries the frequency between marker cursors for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:DV?	Queries all of the settings for level measurement between marker cursors for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:DV:STATe	Sets or queries the on/off status of the level measured between marker cursors for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:DV:VALue?	Queries the level between marker cursors for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>?	Queries all of the settings for frequency measurement of a marker cursor for FFT analysis.	5-95
:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:STATe	Sets or queries the on/off status of the frequency of marker cursor for FFT analysis.	5-96
:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:VALue?	Queries the frequency of a marker cursor for FFT analysis.	5-96
:FFT<x>:MEASure:MARKer[:BASic]:POSition<y>	Sets or queries the position of a marker cursor for FFT analysis.	5-96
:FFT<x>:MEASure:MARKer[:BASic]:V<y>?	Queries all of the settings for level measurement of a marker cursor for FFT analysis.	5-96
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:STATe	Sets or queries the on/off status of the level of marker cursor for FFT analysis.	5-96
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:VALue?	Queries the level of a marker cursor for FFT analysis.	5-96
:FFT<x>:MEASure:MODE	Sets or queries the FFT analysis automatic measurement mode.	5-96
:FFT<x>:MEASure:PEAK?	Queries all peak measurement settings for FFT analysis.	5-96
:FFT<x>:MEASure:PEAK[:BASic]?	Queries all Basic peak item settings for FFT analysis.	5-96
:FFT<x>:MEASure:PEAK[:BASic]:ALL	Collectively turns on or off all Basic peak items for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:DFRequency?	Queries all of the settings for frequency measurement between peak values for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:DFRequency:STATe	Sets or queries the on/off status of the frequency measured between peak values for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:DFRequency:VALue?	Queries the frequency between peak values for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:DV?	Queries all of the settings for power measurement between peak values for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:DV:STATe	Sets or queries the on/off status of the power measured between peak values for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:DV:VALue?	Queries the power between peak values for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>?	Queries all of the settings for a peak frequency for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>:STATe	Sets or queries the on/off status of a peak frequency value for FFT analysis.	5-97
:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>:VALue?	Queries a peak frequency value for FFT analysis.	5-98
:FFT<x>:MEASure:PEAK[:BASic]:RANGe<y>	Sets or queries the measurement range of a peak value for FFT analysis.	5-98
:FFT<x>:MEASure:PEAK[:BASic]:V<y>?	Queries all of the settings for a peak value for FFT analysis.	5-98
:FFT<x>:MEASure:PEAK[:BASic]:V<y>:STATe	Sets or queries the on/off status of a peak value for FFT analysis.	5-98
:FFT<x>:MEASure:PEAK[:BASic]:V<y>:VALue?	Queries a peak value for FFT analysis.	5-98
:FFT<x>:MODE	Sets or queries the waveform display mode for FFT analysis.	5-98
:FFT<x>:RANGe	Sets or queries the measurement source window for FFT analysis.	5-98

Command	Function	Page
:FFT<x>:RPOSition	Sets or queries the center point for magnifying the vertical axis for FFT analysis.	5-98
:FFT<x>:TRACe	Sets or queries the source waveform for FFT analysis.	5-98
:FFT<x>:TYPE	Sets or queries the spectrum for FFT analysis.	5-99
:FFT<x>:UNIT?	Queries all FFT computation unit settings.	5-99
:FFT<x>:UNIT[:DEFine]	Sets or queries an FFT computation unit.	5-99
:FFT<x>:UNIT:MODE	Sets or queries whether an FFT computation unit will be attached automatically or manually.	5-99
:FFT<x>:VERTical?	Queries all vertical axis settings for FFT analysis.	5-99
:FFT<x>:VERTical:LEVel	Sets or queries the vertical axis display position for FFT analysis.	5-99
:FFT<x>:VERTical:MODE	Sets or queries the FFT analysis vertical axis mode.	5-99
:FFT<x>:VERTical:SENSitivity	Sets or queries the vertical sensitivity for FFT analysis.	5-99
:FFT<x>:VTDisplay	Sets or queries the on/off status of the VT waveform display for FFT analysis.	5-99
:FFT<x>:WINDow	Sets or queries the window function for FFT analysis.	5-100

FILE Group

:FILE?	Queries all settings for data storage.	5-101
:FILE:COPY:ABORt	Aborts file copying.	5-101
:FILE:COPY:CDIRectory	Changes the file copy destination directory.	5-101
:FILE:COPY:DRIVe	Sets the file copy source medium.	5-101
:FILE:COPY[:EXECute]	Executes file copying. This is an overlap command.	5-101
:FILE:COPY:PATH?	Queries the file copy destination directory.	5-101
:FILE:DELete:{AHIStogram|ASCii|BINary|BMP|FFT|HLISt|JPEG|MEASure|PNG|SBUS|SETup|SNAP|ZPOLygon|ZWAVe}[:EXECute]	Deletes a specific type of data file. This is an overlap command.	5-101
:FILE[:DIRectory]:CDIRectory	Changes the current directory.	5-101
:FILE[:DIRectory]:DRIVe	Sets the current medium.	5-101
:FILE[:DIRectory]:FREE?	Queries the free space on the current medium in bytes.	5-101
:FILE[:DIRectory]:MDIRectory	Creates a directory in the current directory. This is an overlap command.	5-101
:FILE[:DIRectory]:PATH?	Queries the current directory.	5-102
:FILE:LOAD:BINary:ABORt	Aborts the loading of waveform data.	5-102
:FILE:LOAD:BINary[:EXECute]	Loads waveform data. This is an overlap command.	5-102
:FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}:ABORt	Aborts the loading of a specific type of data.	5-102
:FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}[:EXECute]	Loads a specific type of data. This is an overlap command.	5-102
:FILE:MOVE:ABORt	Aborts file moving.	5-102
:FILE:MOVE:CDIRectory	Changes the file move destination directory.	5-102
:FILE:MOVE:DRIVe	Sets the file move destination medium.	5-102
:FILE:MOVE[:EXECute]	Executes file moving. This is an overlap command.	5-102
:FILE:MOVE:PATH?	Queries the file move destination directory.	5-102
:FILE:PROTect[:EXECute]	Turns file protection on or off.	5-102
:FILE:REName[:EXECute]	Renames a file.	5-102
:FILE:SAVE?	Queries all file save settings.	5-102
:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|SBUS|ZWAVe}?	Queries all of the settings related to the saving of a specified type of file.	5-102
:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}:ABORt	Aborts the saving of a specific type of data.	5-103
:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}[:EXECute]	Saves a specific type of file. This is an overlap command.	5-103
:FILE:SAVE:{AHIStogram|FFT|HARMonics|SBUS|ZWAVe}:SELect	Sets or queries the area of a specific type of data to save.	5-103
:FILE:SAVE:ANAMing	Sets or queries the on/off status of the auto naming feature for saving files.	5-103
:FILE:SAVE:{ASCii|BINary}:COMPression	Sets or queries the save compression method of a specific type of data.	5-103
:FILE:SAVE:{ASCii|BINary}:HISTory	Sets or queries how history waveforms will be saved.	5-103

Command	Function	Page
:FILE:SAVE:{ASCii|BINary}:LENGth	Sets or queries the number of data points to save when compressing or sampling (decimating) various types of data.	5-103
:FILE:SAVE:{ASCii|BINary}:RANGe	Sets or queries the save range for a specific type of data.	5-103
:FILE:SAVE:{ASCii|BINary}:TRACe	Sets or queries the waveforms to save for a specific type of data.	5-103
:FILE:SAVE:ASCii:TINFormation (Time Information)	Sets or queries whether waveform data is saved with time information (ON) or without it (OFF).	5-104
:FILE:SAVE:FFT:FINFormation (Frequency Information)	Sets or queries whether FFT data is saved with frequency information (ON) or without it (OFF).	5-104
:FILE:SAVE:COMMent	Sets or queries the comment that will be attached to the data that will be saved.	5-104
:FILE:SAVE:NAME	Sets or queries the file name for the data that will be saved.	5-104
:FILE:SAVE:SBUS:COMPression	Sets or queries the save compression method of SENT data.	5-104
:FILE:SAVE:SBUS:HISTory	Sets or queries the history for which serial bus data will be saved.	5-104
:FILE:SAVE:SBUS:LENGth	Sets or queries the number of data points to save when compressing or sampling (decimating) SENT data.	5-104
:FILE:SAVE:SBUS:TINFormation (Time Information)	Sets or queries whether SENT waveform data is saved with time information (ON) or without it (OFF).	5-104
:FILE:SAVE:SBUS:TYPe	Sets or queries the save method of SENT data.	5-104

GONogo Group

Command	Function	Page
:GONogo?	Queries all GO/NO-GO determination settings.	5-105
:GONogo:ABORt	Aborts GO/NO-GO determination.	5-105
:GONogo:ACTion?	Queries all of the settings related to the action executed when GO/NO-GO results are NO-GO and the reference.	5-105
:GONogo:ACTion:BUZZer	Sets or queries whether or not the DLM2000 will sound an alarm when a GO/NO-GO result is NO-GO.	5-105
:GONogo:ACTion:HCOPy	Sets or queries whether or not the DLM2000 will print screen images to the optional built-in printer or an external printer for NO-GO results.	5-105
:GONogo:ACTion:MAIL?	Queries all of the settings related to email notification for NO-GO results.	5-105
:GONogo:ACTion:MAIL:COUNt	Sets or queries the upper limit of emails that will be sent for NO-GO results.	5-105
:GONogo:ACTion:MAIL:MODE	Sets or queries whether or not the DLM2000 will send email notification for NO-GO results.	5-105
:GONogo:ACTion:SAVE	Sets or queries whether or not the DLM2000 will save waveform data to the storage medium for NO-GO results.	5-105
:GONogo:COUNt?	Queries the number of performed GO/NO-GO determinations.	5-105
:GONogo:EXECute	Executes GO/NO-GO determination. This is an overlap command.	5-105
:GONogo:LOGic	Sets or queries the GO/NO-GO determination logic.	5-105
:GONogo:NGCount?	Queries the GO/NO-GO determination NO-GO count.	5-105
:GONogo:NGStopcount	Sets or queries the number of NO-GO results at which the DLM2000 will stop performing determinations.	5-106
:GONogo:STOPcount	Sets or queries the number of acquisitions at which the DLM2000 will stop performing determinations.	5-106
:GONogo:WAIT?	Waits for the completion of GO/NO-GO determination by using a timeout timer.	5-106
:GONogo[:ZPARameter]?	Queries all zone and parameter determination settings.	5-106
:GONogo[:ZPARameter]:NUMBer<x>?	Queries all of the settings related to the specified reference standard for zone or parameter determination.	5-106
:GONogo[:ZPARameter]:NUMBer<x>:CAUSe?	Queries whether or not the specified waveform parameter is the cause of a NO-GO zone or parameter determination.	5-106
:GONogo[:ZPARameter]:NUMBer<x>:CONDition	Sets or queries the specified waveform parameter’s reference condition for zone or parameter determination.	5-106
:GONogo[:ZPARameter]:NUMBer<x>:MODE	Sets or queries the specified reference standard mode.	5-106
:GONogo[:ZPARameter]:NUMBer<x>:PARameter?	Queries all parameter settings for the specified reference standard.	5-106
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:ITEM	Sets or queries the specified waveform parameter’s item for parameter determination.	5-107
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:LIMit	Sets or queries the specified waveform parameter’s upper and lower limits for parameter determination.	5-107
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:TRACe	Sets or queries the specified waveform parameter’s source waveform for parameter determination.	5-107
:GONogo[:ZPARameter]:NUMBer<x>:PARameter:VALue?	Queries the specified waveform parameter’s measured value for parameter determination.	5-107

Command	Function	Page
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon?	Queries all polygonal zone determination settings.	5-107
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:HPOSition	Sets or queries the horizontal position that will be used for polygonal zone determination.	5-108
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:RANGe	Sets or queries the source window that will be used for polygonal zone determination.	5-108
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:TRACe	Sets or queries the source waveform that will be used for polygonal zone determination.	5-108
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:VPOSition	Sets or queries the vertical position that will be used for polygonal zone determination.	5-108
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:ZNUMber	Sets or queries the zone number that will be used for polygonal zone determination.	5-108
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle?	Queries all rectangular zone determination settings.	5-108
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:HORizontal	Sets or queries the horizontal position of the rectangle that will be used for rectangular zone determination.	5-109
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:RANGe	Sets or queries the source window of the rectangle that will be used for rectangular zone determination.	5-109
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:TRACe	Sets or queries the source waveform of the rectangle that will be used for rectangular zone determination.	5-109
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:VERTical	Sets or queries the vertical position of the rectangle that will be used for rectangular zone determination.	5-109
:GONogo[:ZPARameter]:NUMBer<x>:WAVE?	Queries all waveform zone determination settings.	5-109
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:EXIT	Exits from the waveform zone edit menu.	5-109
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:NEW	Sets the base waveform in waveform zone editing.	5-109
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:PART	Executes partial editing in waveform zone editing.	5-110
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:EDIT:WHOLe	Executes whole editing in waveform zone editing.	5-110
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:RANGe	Sets or queries the source window that will be used for waveform zone determination.	5-110
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRACe	Sets or queries the source waveform that will be used for waveform zone determination.	5-110
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRANge	Sets or queries the determination area that will be used for waveform zone determination.	5-110
:GONogo[:ZPARameter]:NUMBer<x>:WAVE:ZNUMber	Sets or queries the zone number that will be used for waveform zone determination.	5-110

HCOPy Group

:HCOPy?	Queries all screen capture data output settings.	5-111
:HCOPy:ABORt	Stops data output and paper feeding.	5-111
:HCOPy:COMMent	Sets or queries the comment at the lower right of the screen.	5-111
:HCOPy:DIRection	Sets or queries the data output destination.	5-111
:HCOPy:EXECute	Executes data output.	5-111
:HCOPy:EXTPrinter?	Queries all external printer output settings.	5-111
:HCOPy:EXTPrinter:MODE	Sets or queries whether the DLM2000 will print to the external printer in normal copy or hard copy mode.	5-111
:HCOPy:EXTPrinter:TONE	Sets or queries the external printer output color setting.	5-111
:HCOPy:EXTPrinter:TYPE	Sets or queries the external printer output command type.	5-111
:HCOPy:MULTitarget?	Queries all multi target feature settings.	5-111
:HCOPy:MULTitarget:EXTPrinter	Sets or queries whether the multi target feature will send output to the external printer.	5-111
:HCOPy:MULTitarget:PRINter	Sets or queries whether the multi target feature will send output to the built-in printer.	5-111
:HCOPy:MULTitarget:NETPrinter	Sets or queries whether the multi target feature will send output to a network printer.	5-111
:HCOPy:MULTitarget:FILE	Sets or queries whether the multi target feature will send output to a file.	5-112
:HCOPy:MULTitarget:WAVeform	Sets or queries whether the multi target feature will send output to a waveform file.	5-112
:HCOPy:NETPrint?	Queries all network printer output settings.	5-112

Command	Function	Page
:HCOPy:NETPrint:MODE	Sets or queries whether the DLM2000 will print to the network printer in normal copy or hard copy mode.	5-112
:HCOPy:NETPrint:TONE	Sets or queries the network printer output color setting.	5-112
:HCOPy:NETPrint:TYPE	Sets or queries the network printer output command type.	5-112
:HCOPy:PRINter?	Queries all built-in printer output settings.	5-112
:HCOPy:PRINter:MAG	Sets or queries the magnification used when the DLM2000 prints to the built-in printer in long copy mode.	5-112
:HCOPy:PRINter:MODE	Sets or queries whether the DLM2000 will print to the built-in printer in short, long, or hard copy mode.	5-112
:HCOPy:PRINter:RANGe	Sets or queries the source window used when the DLM2000 prints to the built-in printer in long copy mode.	5-112
:HCOPy:PRINter:REPort	Sets or queries whether the DLM2000 will print additional information on the built-in printer.	5-112
:HCOPy:PRINter:TRANge	Sets or queries the output range used when the DLM2000 prints to the built-in printer in long copy mode.	5-112

HISTory Group

Command	Function	Page
:HISTory?	Queries all of the settings for the history feature.	5-113
:HISTory:AVERage	Sets or queries the highlight display mode for history waveforms.	5-113
:HISTory:DISPlay	Sets or queries the display record start and end numbers.	5-113
:HISTory:DMODe	Sets or queries the history waveform display mode.	5-113
:HISTory:RECord	Sets or queries the history waveform source record.	5-113
:HISTory:RECord? MINimum	Queries the minimum history waveform record number.	5-113
:HISTory:REPLay?	Queries all of the settings for the history waveform replay feature.	5-113
:HISTory:REPLay:JUMP	Jumps to the specified record number in a history waveform.	5-113
:HISTory:REPLay:SPEed	Sets or queries the history waveform replay speed.	5-113
:HISTory:REPLay:STARt	Starts replaying a history waveform in the specified direction.	5-113
:HISTory:REPLay:STOP	Stops history waveform replaying.	5-113
:HISTory[:SEARch]?	Queries all history waveform search settings.	5-114
:HISTory[:SEARch]:ABORt	Aborts searching.	5-114
:HISTory[:SEARch]:EXECute	Executes searching. This is an overlap command.	5-114
:HISTory[:SEARch]:LOGic	Sets or queries the history waveform search logic.	5-114
:HISTory[:SEARch]:RESet	Resets history waveform search conditions.	5-114
:HISTory[:SEARch]:NUMBer<x>?	Queries all of the settings for a search condition.	5-114
:HISTory[:SEARch]:NUMBer<x>:CONDition	Sets or queries the search criterion of a search condition.	5-114
:HISTory[:SEARch]:NUMBer<x>:MODE	Sets or queries the search mode of a search condition.	5-114
:HISTory[:SEARch]:NUMBer<x>:PARameter?	Queries all parameter search settings.	5-114
:HISTory[:SEARch]:NUMBer<x>:PARameter:ITEM	Sets or queries the specified waveform parameter’s item for parameter searching.	5-114
:HISTory[:SEARch]:NUMBer<x>:PARameter:LIMit	Sets or queries the specified waveform parameter’s upper and lower limits for parameter searching.	5-115
:HISTory[:SEARch]:NUMBer<x>:PARameter:TRACe	Sets or queries the specified waveform parameter’s source waveform for parameter searching.	5-115
:HISTory[:SEARch]:NUMBer<x>:PARameter:VALue?	Queries the specified waveform parameter’s measured value for parameter searching.	5-115
:HISTory[:SEARch]:NUMBer<x>:POLYgon?	Queries all polygonal zone search settings.	5-115
:HISTory[:SEARch]:NUMBer<x>:POLYgon:HPOSition	Sets or queries the horizontal position that will be used for polygonal zone searching.	5-115
:HISTory[:SEARch]:NUMBer<x>:POLYgon:RANGe	Sets or queries the source window that will be used for polygonal zone searching.	5-115
:HISTory[:SEARch]:NUMBer<x>:POLYgon:TRACe	Sets or queries the source waveform that will be used for polygonal zone searching.	5-116
:HISTory[:SEARch]:NUMBer<x>:POLYgon:VPOSition	Sets or queries the vertical position that will be used for polygonal zone searching.	5-116
:HISTory[:SEARch]:NUMBer<x>:POLYgon:ZNUMber	Sets or queries the zone number that will be used for polygonal zone searching.	5-116
:HISTory[:SEARch]:NUMBer<x>:RECTangle?	Queries all rectangular zone search settings.	5-116

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

IMAGe Group

:IMAGe?	Queries all screen image data output settings.	5-119
:IMAGe:ABORt	Aborts saving screen image data to the storage medium.	5-119
:IMAGe:BACKground	Sets or queries the screen image background.	5-119
:IMAGe:COMMent	Sets or queries the comment at the lower right of the screen.	5-119
:IMAGe:EXECute	Saves screen image data to a storage medium.	5-119
:IMAGe:FORMat	Sets or queries the screen image output format.	5-119
:IMAGe:INFormation	Sets or queries whether setting information is included in screen capture data.	5-119
:IMAGe:MODE	Sets or queries the screen image output mode.	5-119
:IMAGe:SAVE?	Queries all file output settings.	5-119
:IMAGe:SAVE:ANAMing	Sets or queries the on/off status of the auto naming feature for saving files.	5-119
:IMAGe:SAVE:CDIRectory	Changes the file directory.	5-119
:IMAGe:SAVE:DRIVe	Sets the medium to create files on.	5-119
:IMAGe:SAVE:NAME	Sets or queries the file name for the file that will be created.	5-119
:IMAGe:SEND?	Queries the screen image data value.	5-119
:IMAGe:TONE	Sets or queries the color tone of the screen image data that will be saved.	5-119

INITialize Group

:INITialize:EXECute	Executes initialization.	5-120
:INITialize:UNDO	Undoes initialization.	5-120

LOGic Group

:LOGic?	Queries all logic input waveform settings.	5-121
:LOGic:MODE	Sets or queries the on/off status of logic input.	5-121
:LOGic[:PODA]?	Queries all logic input settings.	5-121
:LOGic[:PODA]:ALL?	Queries all logic input bit settings.	5-121
:LOGic[:PODA]:ALL:DISPlay	Sets the display on/off status of all bits of a specified logic input pod.	5-121
:LOGic[:PODA]:ALL:LEVel	Sets or queries the user-defined threshold level of a specified logic input pod.	5-121
:LOGic[:PODA]:ALL:TYPE	Selects the threshold level of a specified logic input pod.	5-121
:LOGic[:PODA]:BIT<x>?	Queries all of the settings for a bit of a specified logic input pod.	5-121
:LOGic[:PODA]:BIT<x>:DISPlay	Sets the display on/off status of all bits of a specified logic input pod.	5-121
:LOGic[:PODA]:BIT<x>:LABel	Sets or queries the label of a bit of a specified logic input pod.	5-121

Command	Function	Page
:LOGic[:PODA]:BIT<x>:LEVel	Sets or queries the user-defined threshold level for a bit of a specified logic input pod.	5-121
:LOGic[:PODA]:BIT<x>:TYPE	Selects the threshold level for a bit of a specified logic input pod.	5-121
:LOGic[:PODA]:DESKew	Sets or queries the deskewing of a specified logic input pod.	5-122
:LOGic[:PODA]:HYSTeresis	Sets or queries the hysteresis of a specified logic input pod.	5-122
:LOGic:GROup?	Queries all logic settings.	5-122
:LOGic:GROup:BITOrder	Sets or queries the logic bit order.	5-122
:LOGic:GROup:BUNDle?	Queries all logic bundle settings.	5-122
:LOGic:GROup:BUNDle[:MODE]	Sets or queries the logic bundle mode.	5-122
:LOGic:GROup:FORMat	Sets or queries the logic bundle display format (bus display).	5-122
:LOGic:GROup:STATe?	Queries all logic-input state display settings.	5-122
:LOGic:GROup:STATe:CLOCk	Sets or queries the reference clock waveform of the logic-input state display.	5-122
:LOGic:GROup:STATe:HYSTeresis	Sets or queries the hysteresis of the reference clock waveform of the logic-input state display.	5-122
:LOGic:GROup:STATe:MODE	Sets or queries the on/off status of the logic-input state display.	5-122
:LOGic:GROup:STATe:POLarity	Sets or queries the polarity of the reference clock waveform of the logic-input state display.	5-122
:LOGic:GROup:STATe:THReshold	Sets or queries the detection level of the reference clock waveform of the logic-input state display.	5-123
:LOGic:POSition	Sets or queries the vertical position of the logic signal.	5-123
:LOGic:SIZE	Sets or queries the display size of the logic signal.	5-123

MATH Group

:MATH<x>?	Queries all computation settings.	5-124
:MATH<x>:DISPlay	Sets or queries whether or not computed waveforms will be displayed (on/off).	5-124
:MATH<x>:ECOunt?	Queries all edge count settings.	5-124
:MATH<x>:ECOunt:HYSTeresis	Sets or queries the hysteresis for the edge detection level of edge counting.	5-124
:MATH<x>:ECOunt:POLarity	Sets or queries the edge detection polarity for edge counting.	5-124
:MATH<x>:ECOunt:THReshold	Sets or queries the edge detection level for edge-count computation.	5-124
:MATH<x>:FILTer?	Queries all filter settings.	5-124
:MATH<x>:FILTer:FORDer	Sets or queries the filter order of an IIR filter.	5-124
:MATH<x>:FILTer:HCUToff	Sets or queries the cutoff frequency of a high-pass IIR filter.	5-124
:MATH<x>:FILTer:LCUToff	Sets or queries the cutoff frequency of a low-pass IIR filter.	5-124
:MATH<x>:FILTer:TIME	Sets or queries a phase shift delay time.	5-125
:MATH<x>:FILTer:TYPE	Sets or queries a filter type.	5-125
:MATH<x>:FILTer:WEIGht	Sets or queries a moving average weight.	5-125
:MATH<x>:INTegral?	Queries all integration settings.	5-125
:MATH<x>:INTegral:SPOint	Sets or queries the integration start position.	5-125
:MATH<x>:LABel?	Queries all computed waveform label settings.	5-125
:MATH<x>:LABel[:DEFine]	Sets or queries a computed waveform label.	5-125
:MATH<x>:LABel:MODE	Sets or queries the display on/off status of a computed waveform label.	5-125
:MATH<x>:OPERation	Sets or queries an operator.	5-125
:MATH:RCOunt?	Queries all rotary-count computation settings.	5-125
:MATH<x>:RCOunt:THReshold<y>	Sets or queries the threshold level for rotary-count computation.	5-126
:MATH<x>:SCALe?	Queries all scaling settings.	5-126
:MATH<x>:SCALe:CENTer	Sets or queries a center value for manual scaling.	5-126
:MATH<x>:SCALe:MODE	Sets or queries a scaling mode.	5-126
:MATH<x>:SCALe:SENSitivity	Sets or queries a span from the center value for manual scaling.	5-126
:MATH<x>:UNIT?	Queries all computation unit settings.	5-126
:MATH<x>:UNIT[:DEFine]	Sets or queries a unit of computation.	5-126
:MATH<x>:UNIT:MODE	Sets or queries whether a unit of computation will be attached automatically or manually.	5-126
:MATH<x>:USERdefine?	Queries all user-defined computation settings.	5-126
:MATH<x>:USERdefine:AVERage?	Queries all averaging settings for user-defined computation.	5-126
:MATH<x>:USERdefine:AVERage:EWEight	Sets or queries the attenuation constant of exponential averaging in user-defined computation.	5-127
:MATH<x>:USERdefine:AVERage:MODE	Sets or queries the averaging mode for user-defined computation.	5-127
:MATH<x>:USERdefine:CONSitant<y>	Sets or queries a constant for user-defined computation.	5-127

Command	Function	Page
:MATH<x>:USERdefine:DEFine	Sets or queries the equation for user-defined computation.	5-127
:MATH<x>:USERdefine:FILTer<y>?	Queries all filter settings for user-defined computation.	5-127
:MATH<x>:USERdefine:FILTer<y>:BAND	Sets or queries the filter band (frequency band) for user-defined computation.	5-127
:MATH<x>:USERdefine:FILTer<y>:CUToff<z>	Sets or queries the cutoff frequency for user-defined computation.	5-127
:MATH<x>:USERdefine:FILTer<y>:TYPE	Sets or queries the filter type for user-defined computation.	5-128
:MATH<x>:USERdefine:HISTory:ABORt	Aborts user-defined computation (Math on History) of the history waveform.	5-128
:MATH<x>:USERdefine:HISTory:EXECute	Executes user-defined computation (Math on History) of the history waveform.	5-128
:MATH<x>:USERdefine:SCALe?	Queries all scaling settings for user-defined computation.	5-128
:MATH<x>:USERdefine:SCALe:ARANging	Executes auto ranging of user-defined computation.	5-128
:MATH<x>:USERdefine:SCALe:CENTer	Sets or queries the center value for scaling in user-defined computation.	5-128
:MATH<x>:USERdefine:SCALe:SENSitivity	Sets or queries the span from the center value for scaling in user-defined computation.	5-128

MEASure Group

Command	Function	Page
:MEASure?	Queries all of the settings for automated measurement of waveform parameters.	5-129
:MEASure:BIT<x>?	Queries the on/off status of all parameters of a logic waveform.	5-129
:MEASure:BIT<x>:ALL	Collectively turns on or off all measurement items of a logic waveform.	5-129
:MEASure:BIT<x>:AREA2?	Queries all of the on/off status of a parameter of a logic waveform in Area2.	5-129
:MEASure:BIT<x>:AREA2:ALL	Collectively turns on or off all measurement items of a logic waveform in Area2.	5-129
:MEASure:BIT<x>:AREA2:<Parameter>?	Queries the setting of a waveform parameter of a logic waveform in Area2.	5-129
:MEASure:BIT<x>:AREA2:<Parameter>:COUNt?	Queries the statistical processing count of a logic waveform parameter in Area2.	5-129
:MEASure:BIT<x>:AREA2:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a logic waveform parameter in Area2.	5-129
:MEASure:BIT<x>:AREA2:<Parameter>:STATe	Sets or queries the on/off status of a logic waveform parameter in Area2.	5-129
:MEASure:BIT<x>:AREA2:<Parameter>:VALue?	Queries an automatically measured value of a logic waveform parameter in Area2.	5-130
:MEASure:BIT<x>:AREA2:COPY	Copies the on/off status of all measurement items of a logic waveform in Area2 to all other waveforms.	5-130
:MEASure:BIT<x>:AREA2:DELay?	Queries all of the settings for measuring the delay between channels of a logic waveform in Area2.	5-130
:MEASure:BIT<x>:AREA2:DELay:MEASure?	Queries all source waveform settings for measuring the delay between channels of a logic waveform in Area2.	5-130
:MEASure:BIT<x>:AREA2:DELay:MEASure:COUNt	Sets or queries the count number of a source waveform for measuring the delay between channels of a logic waveform in Area2.	5-130
:MEASure:BIT<x>:AREA2:DELay:MEASure:SLOPe	Sets or queries the slope of a source waveform for measuring the delay between channels of a logic waveform in Area2.	5-130
:MEASure:BIT<x>:AREA2:DELay:REFerence?	Queries all reference waveform settings used to measure the delay between the specified logic waveform and the reference waveform in Area2.	5-130
:MEASure:BIT<x>:AREA2:DELay:REFerence:COUNt	Sets or queries the edge count of the reference waveform used to measure the delay between the specified logic waveform and the reference waveform in Area2.	5-131
:MEASure:BIT<x>:AREA2:DELay:REFerence:SLOPe	Sets or queries the slope of the reference waveform used to measure the delay between the specified logic waveform and the reference waveform in Area2.	5-131
:MEASure:BIT<x>:AREA2:DELay:REFerence:SOURce	Sets or queries whether the reference point for measuring the delay between channels of a logic waveform of Area2 will be set to the trigger point or a waveform.	5-131
:MEASure:BIT<x>:AREA2:DELay:REFerence:TRACe	Sets or queries the reference waveform edge used to measure the delay between the specified logic waveform and the reference waveform in Area2.	5-131
:MEASure:BIT<x>:AREA2:DELay:STATe	Sets or queries the on/off status and the display format of a delay parameter in Area2.	5-131

Command	Function	Page
:MEASure:BIT<x>:<Parameter>?	Queries the setting of a waveform parameter of a logic waveform.	5-131
:MEASure:BIT<x>:<Parameter>:COUNt?	Queries the statistical processing count of a logic waveform parameter.	5-131
:MEASure:BIT<x>:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a logic waveform parameter.	5-132
:MEASure:BIT<x>:<Parameter>:STATe	Sets or queries the on/off status of a logic waveform parameter.	5-132
:MEASure:BIT<x>:<Parameter>:VALue?	Queries an automatically measured value of a logic waveform parameter.	5-132
:MEASure:BIT<x>:COPY	Copies the on/off status of all measurement items of a logic waveform to all other waveforms.	5-132
:MEASure:BIT<x>:DELay?	Queries all of the settings for measuring the delay between channels of a logic waveform.	5-132
:MEASure:BIT<x>:DELay:MEASure?	Queries all of the settings for a source waveform for measuring the delay between channels of a logic waveform.	5-132
:MEASure:BIT<x>:DELay:MEASure:COUNt	Sets or queries the count number of a source waveform for measuring the delay between channels of a logic waveform.	5-133
:MEASure:BIT<x>:DELay:MEASure:SLOPe	Sets or queries the slope of a source waveform for measuring the delay between channels of a logic waveform.	5-133
:MEASure:BIT<x>:DELay:REFerence?	Queries all reference waveform settings used to measure the delay between the specified logic waveform and the reference waveform.	5-133
:MEASure:BIT<x>:DELay:REFerence:COUNt	Sets or queries the edge count of the reference waveform used to measure the delay between the specified logic waveform and the reference waveform.	5-133
:MEASure:BIT<x>:DELay:REFerence:SLOPe	Sets or queries the slope of the reference waveform used to measure the delay between the specified logic waveform and the reference waveform.	5-133
:MEASure:BIT<x>:DELay:REFerence: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.	5-133
:MEASure:BIT<x>:DELay:REFerence:TRACe	Sets or queries the reference waveform edge used to measure the delay between the specified logic waveform and the reference waveform.	5-133
:MEASure:BIT<x>:DELay:STATe	Sets or queries the on/off status and the display format of a delay parameter.	5-134
:MEASure:{CHANnel<x>|MATH<x>}?	Queries the on/off status of all parameter of a waveform.	5-134
:MEASure:{CHANnel<x>|MATH<x>}:ALL	Collectively turns on or off all measurement items of a waveform.	5-134
:MEASure:{CHANnel<x>|MATH<x>}:AREA2?	Queries the on/off status of all parameter of a waveform in Area2.	5-134
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:ALL	Collectively turns on or off all measurement items of a waveform in Area2.	5-134
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>?	Queries the setting of a waveform parameter of a waveform in Area2.	5-134
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:COUNt?	Queries the statistical processing count of a waveform parameter in Area2.	5-134
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a waveform parameter in Area2.	5-135
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:STATe	Sets or queries the on/off status of a waveform parameter in Area2.	5-135
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:VALue?	Queries an automatically measured value of a waveform parameter in Area2.	5-135
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:COPY	Copies the on/off status of all measurement items of a waveform in Area2 to all other waveforms.	5-136
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:CYCLe	Sets or queries a cycle mode in Area2.	5-136
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay?	Queries all of the settings for measuring the delay between channels of a waveform in Area2.	5-136
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure?	Queries all of the settings for a source waveform for measuring the delay between channels of a waveform in Area2.	5-136
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:COUNt	Sets or queries the count number of a source waveform for measuring the delay between channels of a waveform in Area2.	5-136
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure:SLOPe	Sets or queries the slope of a source waveform for measuring the delay between channels of a waveform in Area2.	5-136
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence?	Queries all reference waveform settings used to measure the delay between the specified waveform and the reference waveform in Area2.	5-136

Command	Function	Page
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:COUNt	Sets or queries the edge count of the reference waveform used to measure the delay between the specified waveform and the reference waveform in Area2.	5-137
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SLOPe	Sets or queries the slope of the reference waveform used to measure the delay between the specified waveform and the reference waveform in Area2.	5-137
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:SOURce	Sets or queries whether the reference point for measuring the delay between channels of a waveform in Area2 will be set to the trigger point or a waveform.	5-137
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:TRACe	Sets or queries the reference waveform edge used to measure the delay between the specified waveform and the reference waveform in Area2.	5-137
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:STATe	Sets or queries the on/off status and the display format of a delay parameter in Area2.	5-138
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>?	Queries the setting of a waveform parameter of a waveform.	5-138
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:COUNt?	Queries the statistical processing count of a waveform parameter.	5-138
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value of a waveform parameter.	5-138
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:STATe	Sets or queries the on/off status of a waveform parameter.	5-139
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:VALue?	Queries an automatically measured value of a waveform parameter.	5-139
:MEASure:{CHANnel<x>|MATH<x>}:COPY	Copies the on/off status of all measurement items of a waveform to all other waveforms.	5-139
:MEASure:{CHANnel<x>|MATH<x>}:CYCLe	Sets or queries a cycle mode.	5-139
:MEASure:{CHANnel<x>|MATH<x>}:DELay?	Queries all of the settings for measuring the delay between channels of a waveform.	5-140
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure?	Queries all of the settings for a source waveform for measuring the delay between channels of a waveform.	5-140
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:COUNt	Sets or queries the number of edges at which delay between channels will be measured for a waveform.	5-140
:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:SLOPe	Sets or queries a source waveform slope that will be used to measure delay between channels.	5-140
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence?	Queries all reference waveform settings used to measure the delay between the specified waveform and the reference waveform.	5-140
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:COUNt	Sets or queries the edge count of the reference waveform used to measure the delay between the specified waveform and the reference waveform.	5-140
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SLOPe	Sets or queries the slope of the reference waveform used to measure the delay between the specified waveform and the reference waveform.	5-141
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SOURce	Sets or queries whether the reference point for measuring the delay between channels of a waveform in Area2 will be set to the trigger point or a waveform in Area2.	5-141
:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:TRACe	Sets or queries the reference waveform edge used to measure the delay between the specified waveform and the reference waveform.	5-141
:MEASure:{CHANnel<x>|MATH<x>}:DELay:STATe	Sets or queries the on/off status and the display format of a delay parameter.	5-141
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal?	Queries all distal, mesial, and proximal settings.	5-141
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:MODE	Sets or queries the distal, mesial, and proximal point mode setting.	5-141
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:PERCent	Sets or queries the distal, mesial, and proximal points as percentages.	5-142
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:UNIT	Sets or queries the distal, mesial, and proximal points as voltages.	5-142
:MEASure:{CHANnel<x>|MATH<x>}:METHod	Sets or queries the high/low point setting method.	5-142
:MEASure:CONTinuous?	Queries all settings for the normal statistical processing of automatically measured waveform parameters.	5-142
:MEASure:CONTinuous:RESTart	Restarts the normal statistical processing of the automatically measured waveform parameters.	5-142

Command	Function	Page
:MEASure:CONTinuous:TLCHange (Trigger Level Change)	Sets or queries whether normal statistical processing of automatically measured waveform parameters is restarted when the trigger level is changed.	5-142
:MEASure:CYCLe?	Queries all cyclic statistical processing settings for automated measurement.	5-142
:MEASure:CYCLe:ABORt	Aborts cyclic statistical processing for automated measurement.	5-142
:MEASure:CYCLe:EXECute	Executes cyclic statistical processing for automated measurement.	5-142
:MEASure:CYCLe:TRACe	Sets or queries the source waveform used in the cyclic statistical processing for automated measurement.	5-142
:MEASure:HISTory:ABORt	Aborts statistical processing of automatically measured history waveforms.	5-143
:MEASure:HISTory:EXECute	Executes statistical processing of automatically measured history waveforms.	5-143
:MEASure:INDicator	Sets or queries the measurement location indicator.	5-143
:MEASure:MODE	Sets or queries the on/off/statistical processing status of automated measurement.	5-143
:MEASure:RANGe<x>	Sets or queries the measurement source window.	5-143
:MEASure:TRANge<x> (Time Range)	Sets or queries a time range.	5-143
:MEASure:USER<x>?	Queries all automatic measurement settings for a Calc item.	5-143
:MEASure:USER<x>:COUNt?	Queries the statistical processing count of the automatically measured value of a Calc item.	5-143
:MEASure:USER<x>:DEFine	Sets or queries the expression for the automatically measured value of a Calc item.	5-143
:MEASure:USER<x>:{MAXimum|MEAN|MINimum|SDEViation}?	Queries a statistical value that is calculated on the automatically measured value of a Calc item.	5-143
:MEASure:USER<X>:NAME	Sets or queries the name of a Calc item.	5-144
:MEASure:USER<x>:STATe	Sets or queries the on/off status of automated measurement of a Calc item.	5-144
:MEASure:USER<X>:UNIT	Sets or queries the unit of a Calc item.	5-144
:MEASure:USER<x>:VALue?	Queries the automatically measured value of a Calc item.	5-144
:MEASure:WAIT?	Waits for the completion of waveform parameter automated measurement with a set timeout.	5-144

RECall Group

:RECall:SETup<x>:EXECute	Recalls setup data from an internal memory area.	5-145

REFerence Group

:REFerence<x>?	Queries all reference waveform settings.	5-146
:REFerence<x>:DISPlay	Sets or queries the display on/off status of a reference waveform.	5-146
:REFerence<x>:LABel?	Queries all label settings of a reference.	5-146
:REFerence<x>:LABel[:DEFine]	Sets or queries a reference waveform label.	5-146
:REFerence<x>:LABel:MODE	Sets or queries the label display on/off status of a reference.	5-146
:REFerence<x>:LOAD	Loads a reference waveform.	5-146
:REFerence<x>:POSition	Sets or queries the vertical position of a reference waveform.	5-146

SEARch Group

:SEARch?	Queries all waveform search settings.	5-147
:SEARch:ABORt	Aborts searching.	5-147
:SEARch:ASCRoll<x>?	Queries all auto scroll settings.	5-147
:SEARch:ASCRoll<x>:JUMP	Moves the center position of a zoom box to the left or right edge of the main window.	5-147
:SEARch:ASCRoll<x>:SPEed	Sets or queries the auto scroll speed of a zoom box.	5-147
:SEARch:ASCRoll<x>:STARt	Starts auto scrolling.	5-147
:SEARch:ASCRoll<x>:STOP	Stops auto scrolling.	5-147
:SEARch:EDGE?	Queries all edge search settings.	5-147
:SEARch:EDGE:HYSTeresis	Sets or queries the edge search level hysteresis.	5-147
:SEARch:EDGE:LEVel	Sets or queries the edge search level.	5-147
:SEARch:EDGE:SLOPe	Sets or queries the edge search slope.	5-147
:SEARch:EDGE:SOURce	Sets or queries the edge search source waveform.	5-148
:SEARch:EPOint	Sets or queries search end point.	5-148
:SEARch:EXECute	Executes searching. This is an overlap command.	5-148
:SEARch:MARK	Sets or queries the on/off status of detection point marks.	5-148
:SEARch:MAG<x>	Sets or queries the zoom factor of a zoom window.	5-148
:SEARch:POSition<x>	Sets or queries the position of a zoom box.	5-148
:SEARch:PPATtern?	Queries all state or state width search settings.	5-148

Command	Function	Page
:SEARch:PPATtern:BITS?	Queries all bit settings for state or state edge searching.	5-148
:SEARch:PPATtern:BITS:HEXa	Sets the bit search requirement in hexadecimal notation for state or state width searching.	5-148
:SEARch:PPATtern:BITS:PATTern	Sets or queries the bit search requirement in binary notation for state or state width searching.	5-148
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}?	Queries all of the settings of a waveform for state or state edge searching.	5-148
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:HYSTeresis	Sets or queries a source waveform hysteresis for state or state width searching.	5-149
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:LEVel	Sets or queries a source waveform search level for state or state width searching.	5-149
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:PATTern	Sets or queries a source waveform search pattern for state or state width searching.	5-149
:SEARch:PPATtern:CLOCk?	Queries all clock channel settings for state or state width searching.	5-149
:SEARch:PPATtern:CLOCk:HYSTeresis	Sets or queries the clock channel hysteresis for state or state width searching.	5-149
:SEARch:PPATtern:CLOCk:LEVel	Sets or queries the clock channel search level for state or state width searching.	5-149
:SEARch:PPATtern:CLOCk:SLOPe	Sets or queries the clock channel slope for state or state width searching.	5-150
:SEARch:PPATtern:CLOCk:SOURce	Sets or queries the clock channel source waveform state or state width searching.	5-150
:SEARch:PPATtern:LOGic	Sets or queries the state or state width search logic.	5-150
:SEARch:PPATtern:POLarity	Sets or queries the state or state width search polarity.	5-150
:SEARch:PPATtern:TIME<x>	Sets or queries the pulse width setting for state or state width searching.	5-150
:SEARch:PPATtern:TYPE	Sets or queries the search type setting for state or state width searching.	5-150
:SEARch:QUALify?	Queries all edge qualified search settings.	5-150
:SEARch:QUALify:CONDition	Sets or queries the edge qualified search condition.	5-150
:SEARch:SELect?	Sets which detected point to display in the zoom window and queries the zoom position of the detected point.	5-150
:SEARch:SELect? MAXimum	Queries the number of detected points.	5-151
:SEARch:SKIP?	Queries all skip mode settings.	5-151
:SEARch:SKIP:DECimation	Sets or queries the number of detected points that will be skipped in skip mode.	5-151
:SEARch:SKIP:HOLDoff	Sets or queries how long searching will be skipped.	5-151
:SEARch:SKIP:MODE	Sets or queries the skip mode.	5-151
:SEARch:SPOint	Sets or queries search start point.	5-151
:SEARch:TWINdow	Sets or queries which window will display the detected points.	5-151
:SEARch:TYPE	Sets or queries search type.	5-151
:SEARch:WIDTh?	Queries all pulse width search settings.	5-151
:SEARch:WIDTh:HYSTeresis	Sets or queries the pulse width search level hysteresis.	5-151
:SEARch:WIDTh:LEVel	Sets or queries the pulse width search level.	5-151
:SEARch:WIDTh:POLarity	Sets or queries the pulse width search level polarity.	5-151
:SEARch:WIDTh:SOURce	Sets or queries the pulse width search source waveform.	5-152
:SEARch:WIDTh:TIME<x>	Sets or queries the pulse width for pulse width searching.	5-152
:SEARch:WIDTh:TYPE	Sets or queries the pulse width search type.	5-152

SERialbus Group

Command	Function	Page
:SERialbus<x>?	Queries all serial bus signal analysis and search settings.	5-153
:SERialbus<x>:ASETup:ABORt	Aborts auto setup on a serial bus signal.	5-153
:SERialbus<x>:ASETup:EXECute	Executes auto setup on a serial bus signal.	5-153
:SERialbus<x>:CAN?	Queries all CAN bus signal settings.	5-153
:SERialbus<x>:CAN:ANALyze?	Queries all CAN bus signal analysis settings.	5-153
:SERialbus<x>:CAN[:ANALyze]:SETup?	Queries all CAN bus signal analysis bus settings.	5-153
:SERialbus<x>:CAN[:ANALyze]:SETup:BRATe	Sets or queries the CAN bus signal analysis bit rate (data transfer rate).	5-153
:SERialbus<x>:CAN[:ANALyze]:SETup:RECessive	Sets or queries the CAN bus signal analysis recessive level.	5-153
:SERialbus<x>:CAN[:ANALyze]:SETup:SOURce	Sets or queries the CAN bus signal analysis source.	5-154
:SERialbus<x>:CAN[:ANALyze]:SETup:SPOint	Sets or queries the CAN bus signal analysis sample point.	5-154
:SERialbus<x>:CAN:DETail?	Queries all CAN bus signal analysis result list settings.	5-154

Command	Function	Page
:SERialbus<x>:CAN:DETail:DISPlay	Sets or queries the display mode for the CAN bus signal analysis result list.	5-154
:SERialbus<x>:CAN:DETail:LIST:ITEM?	Queries all items that will be displayed in the CAN bus signal analysis result list.	5-154
:SERialbus<x>:CAN:DETail:LIST:VALue?	Queries all of the data for the specified analysis number in the CAN bus signal analysis result list.	5-154
:SERialbus<x>:CAN:SEARch?	Queries all CAN bus signal search settings.	5-154
:SERialbus<x>:CAN:SEARch:ABORt	Aborts the CAN bus signal search.	5-154
:SERialbus<x>:CAN:SEARch:EXECute	Executes a CAN bus signal search.	5-154
:SERialbus<x>:CAN:SEARch:FJUMp:ACK	Jumps to the ACK Field in the CAN bus signal search result.	5-154
:SERialbus<x>:CAN:SEARch:FJUMp:CONTrol	Jumps to the Control Field in the CAN bus signal search result.	5-154
:SERialbus<x>:CAN:SEARch:FJUMp:CRC	Jumps to the CRC Field in the CAN bus signal search result.	5-155
:SERialbus<x>:CAN:SEARch:FJUMp:DATA	Jumps to the Data Field in the CAN bus signal search result.	5-155
:SERialbus<x>:CAN:SEARch:FJUMp:IDENtifier	Jumps to the Identifier in the CAN bus signal search result.	5-155
:SERialbus<x>:CAN:SEARch:FJUMp:SOF	Jumps to the SOF in the CAN bus signal search result.	5-155
:SERialbus<x>:CAN:SEARch:SELect	Sets which detected point to display in the CAN bus signal search zoom window and queries the zoom position of the detected point.	5-155
:SERialbus<x>:CAN:SEARch:SELect? MAXimum	Queries the number of detected points in the CAN bus signal search.	5-155
:SERialbus<x>:CAN:SEARch:SETup?	Queries all CAN bus signal search condition settings.	5-155
:SERialbus<x>:CAN:SEARch:SETup:EFRame?	Queries all CAN bus signal search error settings.	5-155
:SERialbus<x>:CAN:SEARch:SETup:EFRame[:MODE]	Sets or queries the CAN bus signal search error frame setting.	5-155
:SERialbus<x>:CAN:SEARch:SETup:EFRame:CRC	Sets or queries the CAN bus signal search CRC error setting.	5-156
:SERialbus<x>:CAN:SEARch:SETup:EFRame:STUFF	Sets or queries the CAN bus signal search stuff error setting.	5-156
:SERialbus<x>:CAN:SEARch:SETup:IDData?	Queries all ID and data condition settings for CAN bus signal searching.	5-156
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK?	Queries all ACK settings for the ID and data conditions for CAN bus signal searching.	5-156
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:MODE	Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN bus signal searching.	5-156
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:ACK:TYPE	Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-156
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA?	Queries all data settings for the ID and data conditions for CAN bus signal searching.	5-156
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:CONDition	Sets or queries the comparison condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-156
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal searching in decimal notation.	5-157
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DLC	Sets or queries the data length code (DLC), which is one of the ID and data conditions, for CAN bus signal searching.	5-157
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA: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.	5-157
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:HEXa<y>	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal searching in hexadecimal notation.	5-157
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:MODE	Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-157
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA: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.	5-157
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:PATTern<y>	Sets or queries the data value, which is one of the ID and data conditions, for CAN bus signal searching in binary notation.	5-158
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for CAN bus signal searching.	5-158

Command	Function	Page
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:SIGN	Sets or queries whether signed or unsigned data format will be used, which is one of the ID and data conditions, for CAN bus signal searching.	5-158
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier?	Queries all identifier settings for the ID and data conditions for CAN bus signal searching.	5-158
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:HEXa	Sets the ID value, which is one of the ID and data conditions, for CAN bus signal searching in hexadecimal notation.	5-158
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier: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.	5-158
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:MODE	Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-158
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PATTern	Sets or queries the ID pattern, which is one of the ID and data conditions, for CAN bus signal searching in binary notation.	5-159
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CAN bus signal searching.	5-159
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal?	Queries all message and signal settings for the ID and data conditions for CAN bus signal searching.	5-159
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:MESSage:ITEM	Sets the message item, which is one of the ID and data conditions, for CAN bus signal searching.	5-159
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SELect	Sets or queries the message and signal condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-159
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal?	Queries all signal settings for the ID and data conditions for CAN bus signal searching.	5-159
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition	Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN bus signal searching.	5-159
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:DECimal<y>	Sets a signal's reference value, which is one of the ID and data conditions, for CAN bus signal searching in decimal notation.	5-160
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:ITEM	Sets the signal item, which is one of the ID and data conditions, for CAN bus signal searching.	5-160
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:RTR	Sets or queries the RTR value, which is one of the ID and data conditions, for CAN bus signal searching.	5-160
:SERialbus<x>:CAN:SEARch:SETup:MODE	Sets or queries the CAN bus signal search mode setting.	5-160
:SERialbus<x>:CANFD?	Queries all CAN FD bus signal settings.	5-160
:SERialbus<x>:CANFD:ANALyze?	Queries all CAN FD bus signal analysis settings.	5-160
:SERialbus<x>:CANFD[:ANALyze]:SETup?	Queries all CAN FD bus signal analysis bus settings.	5-160
:SERialbus<x>:CANFD[:ANALyze]:SETup:BRATe	Sets or queries the CAN FD bus signal analysis bit rate (data transfer rate).	5-160
:SERialbus<x>:CANFD[:ANALyze]:SETup:DBRate	Sets or queries the CAN FD bus signal analysis data bit rate (data transfer rate of data phase).	5-161
:SERialbus<x>:CANFD[:ANALyze]:SETup:DSPoint	Sets or queries the CAN FD bus signal analysis data phase sample point.	5-161
:SERialbus<x>:CANFD[:ANALyze]:SETup:FDSTandard	Sets or queries whether the CAN FD bus signal analysis is compliant to the ISO standard.	5-161
:SERialbus<x>:CANFD[:ANALyze]:SETup:RECessive	Sets or queries the CAN FD bus signal analysis recessive level.	5-161
:SERialbus<x>:CANFD[:ANALyze]:SETup:SOURce	Sets or queries the CAN FD bus signal analysis source.	5-161
:SERialbus<x>:CANFD[:ANALyze]:SETup:SPOint	Sets or queries the CAN FD bus signal analysis sample point.	5-161
:SERialbus<x>:CANFD:DETail?	Queries all CAN FD bus signal analysis result list settings.	5-162
:SERialbus<x>:CANFD:DETail:DISPlay	Sets or queries the display mode for the CAN FD bus signal analysis result list.	5-162
:SERialbus<x>:CANFD:DETail:LIST:ITEM?	Queries all items that will be displayed in the CAN FD bus signal analysis result list.	5-162
:SERialbus<x>:CANFD:DETail:LIST:VALue?	Queries all the data for the specified analysis number in the CAN FD bus signal analysis result list.	5-162
:SERialbus<x>:CANFD:SEARch?	Queries all CAN FD bus signal search settings.	5-162
:SERialbus<x>:CANFD:SEARch:ABORt	Aborts the CAN FD bus signal search.	5-162
:SERialbus<x>:CANFD:SEARch:EXECute	Executes a CAN FD bus signal search.	5-162

Command	Function	Page
:SERialbus<x>:CANFD:SEARch:FJUMp:ACK	Jumps to the ACK Field in the CAN FD bus signal search result.	5-162
:SERialbus<x>:CANFD:SEARch:FJUMp:CONTrol	Jumps to the Control Field in the CAN FD bus signal search result.	5-162
:SERialbus<x>:CANFD:SEARch:FJUMp:CRC	Jumps to the CRC Field in the CAN FD bus signal search result.	5-162
:SERialbus<x>:CANFD:SEARch:FJUMp:DATA	Jumps to the Data Field in the CAN FD bus signal search result.	5-162
:SERialbus<x>:CANFD:SEARch:FJUMp:IDENtifier	Jumps to the Identifier in the CAN FD bus signal search result.	5-162
:SERialbus<x>:CANFD:SEARch:FJUMp:SOF	Jumps to the SOF in the CAN FD bus signal search result.	5-162
:SERialbus<x>:CANFD:SEARch: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.	5-163
:SERialbus<x>:CANFD:SEARch:SELect? MAXimum	Queries the number of detected points in the CAN FD bus signal search.	5-163
:SERialbus<x>:CANFD:SEARch:SETup?	Queries all CAN FD bus signal search condition settings.	5-163
:SERialbus<x>:CANFD:SEARch:SETup:EFRame?	Queries all CAN FD bus signal search error settings.	5-163
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRC	Sets or queries the CAN FD bus signal search CRC error setting.	5-163
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor?	Queries all CRC error factor settings for CAN FD bus signal searching.	5-163
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:CRCSequence	Sets or queries the CRC sequence (a CRC error factor) for CAN FD bus signal searching.	5-163
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:CRCEFactor:SCOunt	Sets or queries the Stuff Count (a CRC error factor) for CAN FD bus signal searching.	5-163
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:FSTuff	Sets or queries the CAN FD bus signal search fixed stuff error setting.	5-163
:SERialbus<x>:CANFD:SEARch:SETup:EFRame[:MODE]	Sets or queries the CAN FD bus signal search error frame setting.	5-164
:SERialbus<x>:CANFD:SEARch:SETup:EFRame:STUFF	Sets or queries the CAN FD bus signal search stuff error setting.	5-164
:SERialbus<x>:CANFD:SEARch:SETup:FDF:CONDition	Sets or queries the CAN FD bus signal search FDF condition setting.	5-164
:SERialbus<x>:CANFD:SEARch:SETup:IDData?	Queries all ID and data condition settings for CAN FD bus signal searching.	5-164
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK?	Queries all ACK settings for the ID and data conditions for CAN FD bus signal searching.	5-164
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:MODE	Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-164
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:ACK:TYPE	Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-164
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA?	Queries all data settings for the ID and data conditions for CAN FD bus signal searching.	5-164
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:BCOunt	Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-165
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:CONDition	Sets or queries the comparison condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-165
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-165
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in decimal notation.	5-165
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA: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.	5-165
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:HEXa<y>	Sets a reference value, which is one of the ID and data conditions, for CAN FD bus signal searching in hexadecimal notation.	5-165
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:MODE	Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-166
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA: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.	5-166
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:PATTern<y>	Sets or queries the data value, which is one of the ID and data conditions, for CAN FD bus signal searching in binary notation.	5-166

Command	Function	Page
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-166
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA: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.	5-166
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier?	Queries all identifier settings for the ID and data conditions for CAN FD bus signal searching.	5-166
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:HEXa	Sets the ID value, which is one of the ID and data conditions, for CAN FD bus signal searching in hexadecimal notation.	5-166
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier: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.	5-167
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:MODE	Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier: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.	5-167
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal?	Queries all message and signal settings for the ID and data conditions for CAN FD bus signal searching.	5-167
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:MESSage:ITEM	Sets the message item, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SELect	Sets or queries the message and signal condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-167
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal?	Queries all signal settings for the ID and data conditions for CAN FD bus signal searching.	5-168
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition	Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-168
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal: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.	5-168
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal:ITEM	Sets the signal item, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-168
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:RTR	Sets or queries the RTR value, which is one of the ID and data conditions, for CAN FD bus signal searching.	5-168
:SERialbus<x>:CANFD:SEARch:SETup:MODE	Sets or queries the CAN FD bus signal search mode setting.	5-168
:SERialbus<x>:CXPI?	Queries all CXPI bus signal analysis and search settings.	5-169
:SERialbus<x>:CXPI:ANALyze?	Queries all CXPI bus signal analysis bus settings.	5-169
:SERialbus<x>:CXPI[:ANALyze]:SETup?	Queries all CXPI bus signal analysis bus settings.	5-169
:SERialbus<x>:CXPI[:ANALyze]:SETup:BRATe	Sets or queries the CXPI bus signal analysis bit rate (data transfer rate).	5-169
:SERialbus<x>:CXPI[:ANALyze]:SETup:CEDetection	Sets or queries the enable/disable condition of counter error detection for CXPI bus signal analysis.	5-169
:SERialbus<x>:CXPI[:ANALyze]:SETup:CTOLerance	Sets or queries the CXPI bus signal analysis clock tolerance.	5-169
:SERialbus<x>:CXPI[:ANALyze]:SETup:SOURce	Sets or queries the CXPI bus signal analysis source.	5-169
:SERialbus<x>:CXPI[:ANALyze]:SETup:TSAMple	Sets or queries the logic value (1 or 0) determination threshold for CXPI bus signal analysis.	5-170
:SERialbus<x>:CXPI:DETail?	Queries all CXPI bus signal analysis result list settings.	5-170
:SERialbus<x>:CXPI:DETail:DISPlay	Sets or queries the display mode for the CXPI bus signal analysis result list.	5-170
:SERialbus<x>:CXPI:DETail:LIST:ALL?	Queries all the data for all analysis numbers in the CXPI bus signal analysis result list.	5-170
:SERialbus<x>:CXPI:DETail:LIST:ITEM?	Queries all items that will be displayed in the CXPI bus signal analysis result list.	5-170
:SERialbus<x>:CXPI:DETail:LIST:VALue?	Queries all the data for the specified analysis number in the CXPI bus signal analysis result list.	5-170
:SERialbus<x>:CXPI:SEARch?	Queries all CXPI bus signal search settings.	5-170
:SERialbus<x>:CXPI:SEARch:ABORt	Aborts the CXPI bus signal search.	5-170
:SERialbus<x>:CXPI:SEARch:EXECute	Executes a CXPI bus signal search.	5-170

Command	Function	Page
:SERialbus<x>:CXPI:SEARch:SELect	Sets which detected point to display in the CXPI bus signal search zoom window and queries the zoom position of the detected point.	5-170
:SERialbus<x>:CXPI:SEARch:SELect? MAXimum	Queries the number of detected points in the CXPI bus signal search.	5-171
:SERialbus<x>:CXPI:SEARch:SETup?	Queries all CXPI bus signal search condition settings.	5-171
:SERialbus<x>:CXPI:SEARch:SETup:ERRor?	Queries all CXPI bus signal search error settings.	5-171
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:CLOCk	Sets or queries the CXPI bus signal search clock error setting.	5-171
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:COUNter	Sets or queries the CXPI bus signal search counter error setting.	5-171
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:CRC	Sets or queries the CXPI bus signal search CRC error setting.	5-171
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:DLENgth	Sets or queries the CXPI bus signal search data length error setting.	5-171
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:FRAMing	Sets or queries the CXPI bus signal search framing error setting.	5-172
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:IBS	Sets or queries the CXPI bus signal search IBS error setting.	5-172
:SERialbus<x>:CXPI:SEARch:SETup:ERRor:PARity	Sets or queries the CXPI bus signal search parity error setting.	5-172
:SERialbus<x>:CXPI:SEARch:SETup:IDData?	Queries all ID and data condition settings for CXPI bus signal searching.	5-172
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA?	Queries all data settings for the ID and data conditions for CXPI bus signal searching.	5-172
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:BCOunt	Sets or queries the pattern comparison position, which is one of the ID and data conditions, for CXPI bus signal searching.	5-172
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:CONDition	Sets or queries the comparison condition, which is one of the ID and data conditions, for CXPI bus signal searching.	5-172
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for CXPI bus signal searching.	5-173
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in decimal notation.	5-173
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA: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.	5-173
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:HEXa<y>	Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in hexadecimal notation.	5-173
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:MODE	Sets or queries the ID and data condition (enabled/disabled) for CXPI bus signal searching.	5-173
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA: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.	5-173
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:PATTern<y>	Sets a reference value, which is one of the ID and data conditions, for CXPI bus signal searching in binary notation.	5-174
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for CXPI bus signal searching.	5-174
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:SIGN	Sets or queries whether signed or unsigned data format will be used, which is one of the ID and data conditions, for CXPI bus signal searching.	5-174
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation?	Queries all frame information settings for the ID and data conditions for CXPI bus signal searching.	5-174
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:CT	Sets or queries the frame information counter value, which is one of the ID and data conditions, for CXPI bus signal searching.	5-174
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:MODE	Sets or queries the frame information (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal searching.	5-174
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:SLEEP	Sets or queries the frame information sleep bit, which is one of the ID and data conditions, for CXPI bus signal searching.	5-175
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:FINFormation:WAKeup	Sets or queries the frame information wakeup bit, which is one of the ID and data conditions, for CXPI bus signal searching.	5-175
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID?	Queries all ID settings for the ID and data conditions for CXPI bus signal searching.	5-175
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:HEXa	Sets an ID, which is one of the ID and data conditions, for CXPI bus signal searching in hexadecimal notation.	5-175
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:MODE	Sets or queries the ID condition (enabled/disabled), which is one of the ID and data conditions, for CXPI bus signal searching.	5-175

Command	Function	Page
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PATTern	Sets or queries the ID, which is one of the ID and data conditions, for CXPI bus signal searching in binary notation.	5-175
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CXPI bus signal searching.	5-175
:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:ID:PTYPE	Sets or queries the PTYPE comparison condition, which is one of the ID and data conditions, for CXPI bus signal searching.	5-176
:SERialbus<x>:CXPI:SEARch:SETup:MODE	Sets or queries the CXPI bus signal search type.	5-176
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep?	Queries all wakeup and sleep settings for CXPI bus signal searching.	5-176
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:SFRame	Sets or queries the CXPI bus signal search sleep frame setting.	5-176
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:SLEEP	Sets or queries the CXPI bus signal search sleep (clock unavailable condition) setting.	5-176
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:WAKeup	Sets or queries the CXPI bus signal search wakeup (clock available condition) setting.	5-176
:SERialbus<x>:CXPI:SEARch:SETup:WAKeupsleep:WPULse	Sets or queries the CXPI bus signal search wakeup pulse setting.	5-176
:SERialbus<x>:DECode?	Queries all settings related to the decoding of serial bus signals.	5-176
:SERialbus<x>:DECode[:FORMat]	Sets or queries the decode display format of serial bus signals.	5-177
:SERialbus<x>:DECode:SSCMode	Sets or queries the decode display of the start/stop condition of I^{2}C serial bus signals.	5-177
:SERialbus<x>:DISPlay	Sets or queries whether or not serial bus signal analysis will be performed.	5-177
:SERialbus<x>:FLEXray?	Queries all FLEXRAY bus signal settings.	5-177
:SERialbus<x>:FLEXray:ANALyze?	Queries all FLEXRAY bus signal settings.	5-177
:SERialbus<x>:FLEXray[:ANALyze]:SETup?	Queries all FLEXRAY bus signal analysis settings.	5-177
:SERialbus<x>:FLEXray[:ANALyze]:SETup:BCHannel	Sets or queries the channel bus type for FLEXRAY bus signal analysis.	5-177
:SERialbus<x>:FLEXray[:ANALyze]:SETup:BRATe	Sets or queries the FLEXRAY bus signal analysis bit rate (data transfer rate).	5-177
:SERialbus<x>:FLEXray[:ANALyze]:SETup:SOURce	Sets or queries the source signal for FLEXRAY bus signal analysis.	5-177
:SERialbus<x>:FLEXray[:ANALyze]:SETup:SPOint	Sets or queries the FLEXRAY bus signal sample point.	5-178
:SERialbus<x>:FLEXray:DETail?	Queries all FLEXRAY bus signal analysis result list settings.	5-178
:SERialbus<x>:FLEXray:DETail:DISPlay	Sets or queries the display mode for the FLEXRAY bus signal analysis result list.	5-178
:SERialbus<x>:FLEXray:DETail:LIST:ITEM?	Queries all items that will be displayed in the FLEXRAY bus signal analysis result list.	5-178
:SERialbus<x>:FLEXray:DETail:LIST:VALue?	Queries all of the data for the specified analysis number in the FLEXRAY bus signal analysis result list.	5-178
:SERialbus<x>:FLEXray:SEARch?	Queries all FLEXRAY bus signal search settings.	5-178
:SERialbus<x>:FLEXray:SEARch:ABORt	Aborts the FLEXRAY bus signal search.	5-178
:SERialbus<x>:FLEXray:SEARch:EXECute	Executes a FLEXRAY bus signal search.	5-178
:SERialbus<x>:FLEXray:SEARch:FJUMp:CCOunt	Jumps to the cycle-count field in the FLEXRAY bus signal analysis results.	5-178
:SERialbus<x>:FLEXray:SEARch:FJUMp:CRC	Jumps to the CRC Field in the FLEXRAY bus signal analysis results.	5-178
:SERialbus<x>:FLEXray:SEARch:FJUMp:FRAMeid	Jumps to the frame ID field in the FLEXRAY bus signal analysis results.	5-179
:SERialbus<x>:FLEXray:SEARch:FJUMp:HCRC	Jumps to the Header CRC Field in the FLEXRAY bus signal analysis results.	5-179
:SERialbus<x>:FLEXray:SEARch:FJUMp:PLENgth	Jumps to the Payload Length Field in the FLEXRAY bus signal analysis results.	5-179
:SERialbus<x>:FLEXray:SEARch:SELect	Sets the detected-waveform number of the FLEXRAY bus signal search, or queries the zoom location that corresponds to the number.	5-179
:SERialbus<x>:FLEXray:SEARch:SETup?	Queries all FLEXRAY bus signal search condition settings.	5-179
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor?	Queries all FLEXRAY bus signal search error settings.	5-179

Command	Function	Page
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:BSS	Sets or queries the FLEXRAY bus signal search BSS error setting.	5-179
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:CRC	Sets or queries the FLEXRAY bus signal search CRC error setting.	5-179
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:FES	Sets or queries the FLEXRAY bus signal search FES error setting.	5-179
:SERialbus<x>:FLEXray:SEARch:SETup:ERRor:HCRC	Sets or queries the FLEXRAY bus signal search Header CRC error setting.	5-180
:SERialbus<x>:FLEXray:SEARch:SETup:IDData?	Queries all ID and data condition settings for FLEXRAY bus signal searching.	5-180
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt?	Queries all cycle-count settings for FLEXRAY bus signal searching.	5-180
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition	Sets or queries the cycle-count data-condition settings for FLEXRAY bus signal searching.	5-180
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt<y>	Sets or queries a FLEXRAY bus signal-search cycle-count setting.	5-180
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:MODE	Sets or queries the cycle-count mode, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-180
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA?	Queries all ID and data condition settings for FLEXRAY bus signal searching.	5-181
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:BCOunt	Sets or queries the position for comparing data patterns, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-181
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:CONDition	Sets or queries the comparison condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-181
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:DBYTe	Sets or queries the number of data bytes, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-181
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA: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.	5-181
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA: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.	5-181
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:HEXa<y>	Sets a reference value, which is one of the ID and data conditions, for FLEXRAY bus signal searching in hexadecimal notation.	5-182
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MODE	Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-182
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA: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.	5-182
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:PATTern<y>	Sets or queries the data value, which is one of the ID and data conditions, for FLEXRAY bus signal searching in binary notation.	5-182
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-182
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:SIGN	Sets or queries whether a signed or unsigned data format will be used (this is one of the ID and data conditions) for CAN bus signal searching.	5-182
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID?	Queries all frame ID settings for the ID and data conditions for FLEXRAY bus signal searching.	5-183
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:CONDition	Sets or queries the frame ID data condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:ID<y>	Sets or queries the frame ID value, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:MODE	Sets or queries the frame ID enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator?	Queries all indicator settings for the ID and data conditions for FLEXRAY bus signal searching.	5-183
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:MODE	Sets or queries the indicator enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:NFRame	Sets or queries the indicator null frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-183
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:PPReamble	Sets or queries the indicator payload preamble, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:STFRame	Sets or queries whether or not the indicator start frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:SYFRame	Sets or queries the indicator sync frame, which is one of the ID and data conditions, for FLEXRAY bus signal searching.	5-184

Command	Function	Page
:SERialbus<x>:FLEXray:SEARch:SETup:MODE	Sets or queries the FLEXRAY bus signal search mode.	5-184
:SERialbus<x>:I2C?	Queries all I^{2}C bus signal analysis and search settings.	5-184
:SERialbus<x>:I2C:ANALyze?	Queries all I^{2}C bus signal analysis settings.	5-184
:SERialbus<x>:I2C[:ANALyze]:SETup?	Queries all I^{2}C bus signal analysis bus settings.	5-184
:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk?	Queries all I^{2}C bus signal analysis clock settings.	5-184
:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk:SOURce	Sets or queries the clock source for I^{2}C bus signal analysis.	5-184
:SERialbus<x>:I2C[:ANALyze]:SETup:DATA?	Queries all I^{2}C bus signal analysis data settings.	5-185
:SERialbus<x>:I2C[:ANALyze]:SETup:DATA:SOURce	Sets or queries the I^{2}C bus signal data source.	5-185
:SERialbus<x>:I2C:DETail?	Queries all I^{2}C bus signal analysis result list settings.	5-185
:SERialbus<x>:I2C:DETail:DISPlay	Sets or queries the display mode for the I^{2}C bus signal analysis result list.	5-185
:SERialbus<x>:I2C:DETail:LIST:ITEM?	Queries all items that will be displayed in the I^{2}C bus signal analysis result list.	5-185
:SERialbus<x>:I2C:DETail:LIST:VALue?	Queries all of the data for the specified analysis number in the I^{2}C bus signal analysis result list.	5-185
:SERialbus<x>:I2C[:ANALyze]:SETup:INCLuderw	Sets or queries the on/off status of the R/W address bit in I^{2}C bus signal analysis.	5-185
:SERialbus<x>:I2C:SEARch?	Queries all I^{2}C bus signal search settings.	5-185
:SERialbus<x>:I2C:SEARch:ABORt	Aborts the I^{2}C bus signal search.	5-185
:SERialbus<x>:I2C:SEARch:EXECute	Executes a I^{2}C bus signal search.	5-185
:SERialbus<x>:I2C:SEARch:SELect	Sets which detected point to display in the I^{2}C bus signal search zoom window and queries the zoom position of the detected point.	5-186
:SERialbus<x>:I2C:SEARch:SELect? MAXimum	Queries the number of detected points in the I^{2}C bus signal search.	5-186
:SERialbus<x>:I2C:SEARch:SETup?	Queries all I^{2}C bus signal search condition settings.	5-186
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess?	Queries all address pattern settings for I^{2}C bus signal searching.	5-186
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess?	Queries all address pattern address settings for I^{2}C bus signal searching.	5-186
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress?	Queries all 10-bit address settings for I^{2}C bus signal searching.	5-186
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:HEXa	Sets the 10-bit address for I^{2}C bus signal searching in hexadecimal notation.	5-186
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:PATTern	Sets or queries the 10-bit address for I^{2}C bus signal searching in binary notation.	5-186
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]?	Queries all 7-bit address settings for I^{2}C bus signal searching.	5-186
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:HEXa	Sets the 7-bit address for I^{2}C bus signal searching in hexadecimal notation.	5-187
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:PATTern	Sets or queries the 7-bit address for I^{2}C bus signal searching in binary notation.	5-187
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub?	Queries all 7-bit + sub address settings for I^{2}C bus signal searching.	5-187
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess?	Queries all 7-bit address settings for the 7-bit + sub address for I^{2}C bus signal searching.	5-187
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:HEXa	Sets the 7-bit address for the 7-bit + sub address for I^{2}C bus signal searching in hexadecimal notation.	5-187
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern	Sets or queries the 7-bit address for the 7-bit + sub address for I^{2}C bus signal searching in binary notation.	5-187

Command	Function	Page
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress?	Queries all sub address settings for the 7-bit + sub address for I^{2}C bus signal searching.	5-187
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress:HEXa	Sets the sub address for the 7-bit + sub address for I^{2}C bus signal searching in hexadecimal notation.	5-187
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern	Sets or queries the sub address for the 7-bit + sub address for I^{2}C bus signal searching in binary notation.	5-188
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:MODE	Sets or queries the address condition enable/disable mode for I^{2}C bus signal searching.	5-188
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:PFORmat	Sets or queries the address input format, which is one of the address conditions, for I^{2}C bus signal searching.	5-188
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:TYPE	Sets or queries the address type, which is one of the address conditions, for I^{2}C bus signal searching.	5-188
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA?	Queries all address pattern data settings for I^{2}C bus signal searching.	5-188
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BCOunt	Sets or queries the position for comparing data patterns for I^{2}C bus signal searching.	5-188
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:BMODe	Sets or queries the on/off status of the position for comparing data patterns for I^{2}C bus signal searching.	5-188
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:CONDition	Sets or queries the data comparison condition (true or false) for I^{2}C bus signal searching.	5-189
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:DBYTe	Sets or queries the number of data bytes that will be compared for I^{2}C bus signal searching.	5-189
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:HEXa<y>	Sets the data value, which is one of the data conditions, for I^{2}C bus signal searching in hexadecimal notation.	5-189
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:MODE	Sets or queries the data condition enable/disable mode for I^{2}C bus signal searching.	5-189
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:PATTern<y>	Sets or queries the data value, which is one of the data conditions, for I^{2}C bus signal searching in binary notation.	5-189
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:PFORmat	Sets or queries the data input format, which is one of the data conditions, for I^{2}C bus signal searching.	5-189
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall?	Queries all general call settings for I^{2}C bus signal searching.	5-189
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress?	Queries all general call's 7-bit master address settings for I^{2}C bus signal searching.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress:HEXa	Sets the general call's 7-bit master address for I^{2}C bus signal searching in hexadecimal notation.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress:PATTern	Sets or queries the general call's 7-bit master address for I^{2}C bus signal searching in binary notation.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:SBYTe	Sets or queries the general call's second byte type for I^{2}C bus signal searching.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:MODE	Sets or queries the I^{2}C bus signal search type.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:NONack?	Queries all NON ACK ignore mode settings for I^{2}C bus signal searching.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:HSMode	Sets or queries whether or not NON ACK will be ignored in high-speed mode for I^{2}C bus signal searching.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:READaccess	Sets or queries whether or not NON ACK will be ignored in read access mode for I^{2}C bus signal searching.	5-190
:SERialbus<x>:I2C:SEARch[:SETup]:NONack:STARtbyte	Sets or queries whether or not NON ACK will be ignored in start bytes for I^{2}C bus signal searching.	5-191
:SERialbus<x>:LIN?	Queries all LIN bus signal analysis and search settings.	5-191
:SERialbus<x>:LIN:ANALyze?	Queries all LIN bus signal analysis settings.	5-191
:SERialbus<x>:LIN[:ANALyze]:SETup?	Queries all LIN bus signal analysis bus settings.	5-191
:SERialbus<x>:LIN[:ANALyze]:SETup:BRATe	Sets or queries the LIN bus signal analysis bit rate (data transfer rate).	5-191
:SERialbus<x>:LIN[:ANALyze]:SETup:REVision	Sets or queries the LIN bus signal analysis revision number.	5-191
:SERialbus<x>:LIN[:ANALyze]:SETup:SOURce	Sets or queries the LIN bus signal analysis source.	5-191

Command	Function	Page
:SERialbus<x>:LIN[:ANALyze]:SETup:SPOint	Sets or queries the LIN bus signal sample point.	5-191
:SERialbus<x>:LIN:DETail?	Queries all LIN bus signal analysis result list settings.	5-192
:SERialbus<x>:LIN:DETail:DISPlay	Sets or queries the display mode for the LIN bus signal analysis result list.	5-192
:SERialbus<x>:LIN:DETail:LIST:ITEM?	Queries all items that will be displayed in the LIN bus signal analysis result list.	5-192
:SERialbus<x>:LIN:DETail:LIST:VALue?	Queries all of the data for the specified analysis number in the LIN bus signal analysis result list.	5-192
:SERialbus<x>:LIN:SEARch?	Queries all LIN bus signal search settings.	5-192
:SERialbus<x>:LIN:SEARch:ABORt	Aborts the LIN bus signal search.	5-192
:SERialbus<x>:LIN:SEARch:EXECute	Executes a LIN bus signal search.	5-192
:SERialbus<x>:LIN:SEARch:FJUMp:BREak	Jumps to the break field in the LIN bus signal search result.	5-192
:SERialbus<x>:LIN:SEARch:FJUMp:CSUM	Jumps to the checksum field in the LIN bus signal search result.	5-192
:SERialbus<x>:LIN:SEARch:FJUMp:DATA	Jumps to the data field in the LIN bus signal search result.	5-192
:SERialbus<x>:LIN:SEARch:FJUMp:IDENtifier	Jumps to the identifier field in the LIN bus signal search result.	5-192
:SERialbus<x>:LIN:SEARch:FJUMp:SYNCh	Jumps to the sync field in the LIN bus signal search result.	5-192
:SERialbus<x>:LIN:SEARch:SELect	Sets which detected point to display in the LIN bus signal search zoom window and queries the zoom position of the detected point.	5-193
:SERialbus<x>:LIN:SEARch:SELect? MAXimum	Queries the number of detected points in the LIN bus signal search.	5-193
:SERialbus<x>:LIN:SEARch:SETup?	Queries all LIN bus signal search condition settings.	5-193
:SERialbus<x>:LIN:SEARch:SETup:ERRor?	Queries all LIN bus signal search error settings.	5-193
:SERialbus<x>:LIN:SEARch:SETup:ERRor:CHECksum	Sets or queries the LIN bus signal search checksum error setting.	5-193
:SERialbus<x>:LIN:SEARch:SETup:ERRor:FRAMing	Sets or queries the LIN bus signal search framing error setting.	5-193
:SERialbus<x>:LIN:SEARch:SETup:ERRor:PARity	Sets or queries the LIN bus signal search parity error setting.	5-193
:SERialbus<x>:LIN:SEARch:SETup:ERRor:SYNCh	Sets or queries the LIN bus signal search synch error setting.	5-193
:SERialbus<x>:LIN:SEARch:SETup:ERRor:TIMeout	Sets or queries the LIN bus signal search timeout error setting.	5-194
:SERialbus<x>:LIN:SEARch:SETup:IDData?	Queries all ID and data condition settings for LIN bus signal searching.	5-194
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA?	Queries all LIN bus signal search data settings.	5-194
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:CONDition	Sets or queries the data conditions for LIN bus signal searching.	5-194
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DBYTe	Sets or queries the number of data bytes for LIN bus signal searching.	5-194
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:DECimal<y>	Sets or queries the data value for LIN bus signal searching in decimal notation.	5-194
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:ENDian	Sets or queries the data endian setting for LIN bus signal searching.	5-194
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:HEXa<y>	Sets the data for LIN bus signal searching in hexadecimal notation.	5-195
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MODE	Sets or queries the data enable/disable condition for LIN bus signal searching.	5-195
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MSBLsb	Sets or queries the data MSB and LSB bits for LIN bus signal searching.	5-195
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:PATTern<y>	Sets or queries the data value for LIN bus signal searching in binary notation.	5-195
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:PFORmat	Sets or queries the input format, which is one of the data conditions, for LIN bus signal searching.	5-195
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:SIGN	Sets or queries whether signed or unsigned data format will be used, which is one of the data conditions, for LIN bus signal searching.	5-195

Command	Function	Page
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier?	Queries all identifier settings for the ID and data conditions for LIN bus signal searching.	5-195
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID?	Queries all LIN bus signal search ID settings.	5-196
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:HEXa	Sets the ID for LIN bus signal searching in hexadecimal notation.	5-196
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:MODE	Sets or queries the ID enable/disable condition for LIN bus signal searching.	5-196
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:PATTern	Sets or queries the ID value for LIN bus signal searching in binary notation.	5-196
:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:PFORmat	Sets or queries the ID input format for LIN bus signal searching.	5-196
:SERialbus<x>:LIN:SEARch:SETup:MODE	Sets or queries the LIN bus signal search mode setting.	5-196
:SERialbus<x>:PSI5?	Queries all PSI5 signal analysis and search settings.	5-196
:SERialbus<x>:PSI5:ANALyze?	Queries all PSI5 signal analysis settings.	5-196
:SERialbus<x>:PSI5[:ANALyze]:SETup?	Queries all bus setup settings for PSI5 signal analysis.	5-196
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA?	Queries all PSI5 signal analysis data signal settings.	5-196
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:BRATe	Sets or queries the PSI5 signal analysis bit rate (data transfer rate).	5-197
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:CTOLerance	Sets or queries the PSI5 signal analysis clock tolerance.	5-197
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:DBITs	Sets or queries the data length for PSI5 signal analysis.	5-197
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:EDETection	Sets or queries the error detection method for PSI5 signal analysis.	5-197
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection	Queries all PSI5 signal analysis noise rejection settings.	5-197
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:ETIMe	Sets or queries the rejection end time for PSI5 signal analysis noise rejection.	5-197
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:MODE	Sets or queries the on/off status of the PSI5 signal analysis noise rejection.	5-197
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SOURce	Sets or queries the data source for PSI5 signal analysis.	5-198
:SERialbus<x>:PSI5[:ANALyze]:SETup:NUMBerofslot	Sets or queries the number of slots for PSI5 signal analysis.	5-198
:SERialbus<x>:PSI5[:ANALyze]:SETup:SLOT<y>	Sets or queries the slot’s start and end times for PSI5 signal analysis.	5-198
:SERialbus<x>:PSI5[:ANALyze]:SETup:SYNC	Sets or queries the sync signal for PSI5 signal analysis.	5-198
:SERialbus<x>:PSI5:DETail?	Queries all PSI5 signal analysis list settings.	5-198
:SERialbus<x>:PSI5:DETail:DISPlay	Sets or queries the display mode for the PSI5 signal analysis list.	5-198
:SERialbus<x>:PSI5:DETail:LIST:ALL?	Queries all the data for all analysis numbers in the PSI5 signal analysis result list.	5-199
:SERialbus<x>:PSI5:DETail:LIST:ITEM?	Queries all items that will be displayed in the PSI5 signal analysis result list.	5-199
:SERialbus<x>:PSI5:DETail:LIST:VALue?	Queries all the data for the specified analysis number in the PSI5 signal analysis result list.	5-199
:SERialbus<x>:PSI5:SEARch?	Queries all PSI5 signal search settings.	5-199
:SERialbus<x>:PSI5:SEARch:ABORt	Aborts the PSI5 signal search.	5-199
:SERialbus<x>:PSI5:SEARch:EXECute	Executes a PSI5 signal search.	5-199
:SERialbus<x>:PSI5:SEARch:SELect	Sets which detected point to display in the PSI5 signal search zoom window and queries the zoom position of the detected point.	5-199
:SERialbus<x>:PSI5:SEARch:SETup?	Queries all PSI5 signal search condition settings.	5-199
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA?	Queries all PSI5 signal data search settings.	5-199
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA?	Queries all data settings of the PSI5 signal data search.	5-199

Command	Function	Page
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:CONDition	Sets or queries the comparison condition of the PSI5 signal data search.	5-199
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:DECimal<y>	Sets or queries the data of the PSI5 signal data search in decimal notation.	5-200
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:HEXa	Sets the data of the PSI5 signal data search in hexadecimal notation.	5-200
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:PATTern	Sets or queries the data of the PSI5 signal data search in binary notation.	5-200
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:PFORmat	Sets or queries the data input format of the PSI5 signal data search in binary notation.	5-200
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:FRAMeinslot?	Queries all slot specification settings of the PSI5 signal data search.	5-200
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:FRAMeinslot:MODE	Sets or queries the slot specification condition (enabled/disabled) of the PSI5 signal data search.	5-200
:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:FRAMeinslot:SNUMber	Sets or queries the slot number of the PSI5 signal data search.	5-200
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor?	Queries all PSI5 signal error search settings.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:CLOCk	Sets or queries the PSI5 signal clock error search.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FNUMber	Sets or queries the PSI5 signal frame number error search.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FRAMe	Sets or queries the PSI5 signal frame error search.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:PCRC	Sets or queries the PSI5 signal parity/CRC error search.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBIT	Sets or queries the PSI5 signal start bit error search.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBOundary	Sets or queries the PSI5 signal slot boundary error search.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:FRAMeinslot?	Queries all PSI5 signal slot specification search settings.	5-201
:SERialbus<x>:PSI5:SEARch[:SETup]:FRAMeinslot:SNUMber	Sets or queries the slot number of the PSI5 signal slot specification search.	5-202
:SERialbus<x>:PSI5:SEARch[:SETup]:MODE	Sets or queries the PSI5 signal search mode setting.	5-202
:SERialbus<x>:PSI5:TRENd<y>?	Queries all PSI5 signal analysis trend display settings.	5-202
:SERialbus<x>:PSI5:TRENd<y>:ASCale	Executes auto scaling of the PSI5 signal analysis trend display.	5-202
:SERialbus<x>:PSI5:TRENd<y>:CURSor?	Queries all PSI5 signal analysis trend display cursor measurement settings.	5-202
:SERialbus<x>:PSI5:TRENd<y>:CURSor:DT:VALue?	Queries the time value between cursors on the PSI5 signal analysis trend display.	5-202
:SERialbus<x>:PSI5:TRENd<y>:CURSor:DV:VALue?	Queries the vertical value between cursors on the PSI5 signal analysis trend display.	5-202
:SERialbus<x>:PSI5:TRENd<y>:CURSor:MODE	Sets or queries the cursor mode on the PSI5 signal analysis trend display.	5-202
:SERialbus<x>:PSI5:TRENd<y>:CURSor:POSition<z>	Sets or queries the position of the specified cursor on the PSI5 signal analysis trend display.	5-202
:SERialbus<x>:PSI5:TRENd<y>:CURSor:T<z>:VALue?	Queries the time value at the specified cursor on the PSI5 signal analysis trend display.	5-203
:SERialbus<x>:PSI5:TRENd<y>:CURSor:V<z>:VALue?	Queries the vertical value at the specified cursor on the PSI5 signal analysis trend display.	5-203
:SERialbus<x>:PSI5:TRENd<y>:DISPlay	Sets or queries the on/off status of the PSI5 signal analysis trend display.	5-203
:SERialbus<x>:PSI5:TRENd<y>:HRANge	Sets or queries the PSI5 signal analysis trend display source window.	5-203
:SERialbus<x>:PSI5:TRENd<y>:SOURce	Sets or queries the target slot number of the PSI5 signal analysis trend display.	5-203
:SERialbus<x>:PSI5:TRENd<y>:VERTical	Sets or queries the vertical range of the PSI5 signal analysis trend display.	5-203
:SERialbus<x>:PSI5:TRENd<y>:VTDisplay	Sets or queries the on/off status of the VT waveform display on the PSI5 signal analysis trend display.	5-203

Command	Function	Page
:SERialbus<x>:RWINdow	Sets or queries in which zoom window, ZOOM1 or ZOOM2, the detected section will be displayed.	5-203
:SERialbus<x>:SENT?	Queries all SENT signal analysis and search settings.	5-203
:SERialbus<x>:SENT:ANALyze?	Queries all SENT signal analysis settings.	5-203
:SERialbus<x>:SENT[:ANALyze]:SETup?	Queries all bus setup settings for SENT signal analysis.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:DISPlay	Sets or queries the display mode for the SENT signal analysis result.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST?	Queries all SENT signal analysis fast channel settings.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:DTYPe	Sets or queries the SENT signal analysis fast channel data format.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup?	Queries all settings related to user-defined data of the SENT signal analysis fast channel.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>?	Queries all settings related to the specified user-defined data of the SENT signal analysis fast channel.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:MODE	Sets or queries the enable/disable condition of the specified user-defined data of the SENT signal analysis fast channel.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:ORDer	Sets or queries the byte order of the specified user-defined data of the SENT signal analysis fast channel.	5-204
:SERialbus<x>:SENT[:ANALyze]:SETup:FAST:USETup:DATA<y>:SIZE	Sets or queries the data size of the specified user-defined data of the SENT signal analysis fast channel.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat?	Queries all SENT signal analysis format settings.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor?	Queries all SENT signal analysis error factor settings.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SAComm?	Queries all status and communication error factor settings of SENT signal analysis.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SAComm:BIT<y>?	Queries all status and communication error factor bit settings of SENT signal analysis.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CEFactor:SCPulses	Sets or queries the consecutive calibration pulse error factor of SENT signal analysis.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CRCType	Sets or queries the SENT signal analysis CRC computation type.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTICk	Sets or queries the SENT signal analysis clock tick value.	5-205
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:CTOLerance	Sets or queries the SENT signal analysis clock tolerance.	5-206
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:DNIBbles	Sets or queries the number of data nibbles for SENT signal analysis.	5-206
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:PPULse	Sets or queries the presence or absence of pause pulses for SENT signal analysis.	5-206
:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat:VERSion	Sets or queries the SENT signal analysis specification version.	5-206
:SERialbus<x>:SENT[:ANALyze]:SETup:STYPe	Sets or queries the SENT signal analysis slow channel format.	5-206
:SERialbus<x>:SENT[:ANALyze]:SETup:SOURce	Sets or queries the SENT signal analysis source.	5-206
:SERialbus<x>:SENT:DETail?	Queries all SENT signal analysis list settings.	5-207
:SERialbus<x>:SENT:DETail:DISPlay	Sets or queries the display mode for the SENT signal analysis list.	5-207
:SERialbus<x>:SENT:DETail:LIST:ALL?	Queries all the data for all analysis numbers in the SENT signal analysis result list.	5-207
:SERialbus<x>:SENT:DETail:LIST:ITEM?	Queries all items that will be displayed in the SENT signal analysis result list.	5-207
:SERialbus<x>:SENT:DETail:LIST:VALue?	Queries all the data for the specified analysis number in the SENT signal analysis result list.	5-207
:SERialbus<x>:SENT:SEARch?	Queries all SENT signal search settings.	5-207
:SERialbus<x>:SENT:SEARch:ABORt	Aborts the SENT signal search.	5-207
:SERialbus<x>:SENT:SEARch:EXECute	Executes a SENT signal search.	5-207
:SERialbus<x>:SENT:SEARch:SELect	Sets which detected point to display in the SENT signal search zoom window and queries the zoom position of the detected point.	5-207

Command	Function	Page
:SERialbus<x>:SENT:SEARch:SELect? MAXimum	Queries the number of detected points in the SENT signal search.	5-208
:SERialbus<x>:SENT:SEARch:SETup?	Queries all SENT signal search condition settings.	5-208
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor?	Queries all SENT signal search error settings.	5-208
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:FCRC	Sets or queries the SENT signal search fast channel CRC error.	5-208
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NDValue	Sets or queries the SENT signal search nibble data value error.	5-208
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:NNUMber	Sets or queries the SENT signal search nibble data count error.	5-208
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SAComm	Sets or queries the SENT signal search status and communication error.	5-208
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCRC	Sets or queries the SENT signal search slow channel CRC error.	5-209
:SERialbus<x>:SENT:SEARch[:SETup]:ERRor:SCPulses	Sets or queries the SENT signal search consecutive calibration pulse error.	5-209
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa?	Queries all fast channel data settings for SENT signal searching.	5-209
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>?	Queries all settings related to the specified user data of the SENT signal search fast channel.	5-209
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:CONDition	Sets or queries the comparison condition of the specified user data of the SENT signal search fast channel.	5-209
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:DECimal<z>	Sets or queries the data of the specified user data of the SENT signal search fast channel in decimal notation.	5-209
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DATA<y>:MODE	Sets or queries the data enable/disable condition of the specified user data of the SENT signal search fast channel.	5-209
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles?	Queries all fast channel nibble data settings for SENT signal searching.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:CONDition	Sets or queries the nibble data comparison condition of the SENT signal search fast channel.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:HEXa	Sets the SENT signal search fast channel nibble data in hexadecimal notation.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:PATTern	Sets or queries the SENT signal search fast channel nibble data in binary notation.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:PFORmat	Sets or queries the data input format, which is one of the fast channel nibble data conditions, for SENT signal searching.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm?	Queries all fast channel status and communication nibble settings for SENT signal searching.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:HEXa	Sets the SENT signal search fast channel status and communication nibble data in hexadecimal notation.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:PATTern	Sets or queries the SENT signal search fast channel status and communication nibble data in binary notation.	5-210
:SERialbus<x>:SENT:SEARch[:SETup]:FSAComm:PFORmat	Sets or queries the data input format, which is one of the fast channel status and communication nibble data conditions, for SENT signal searching.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:MODE	Sets or queries the SENT signal search mode.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa?	Queries all slow channel data settings for SENT signal searching.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced?	Queries all slow channel enhanced type settings for SENT signal searching.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:CBIT	Sets or queries the SENT signal search slow channel enhanced type configuration bit.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit?	Queries all slow channel enhanced type 12-bit data and 8-bit ID settings for SENT signal searching.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA?	Queries all slow channel enhanced type 12-bit data and 8-bit ID data settings for SENT signal searching.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit: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.	5-211
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA: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.	5-212

Command	Function	Page
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:HEXa	Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in hexadecimal notation.	5-212
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA: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.	5-212
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:PATTern	Sets the slow channel enhanced type 12-bit data and 8-bit ID data for SENT signal searching in binary notation.	5-212
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:PFORmat	Sets or queries the data input format, which is one of the slow channel enhanced type 12-bit data and 8-bit ID data conditions, for SENT signal searching.	5-212
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID?	Queries all ID settings related to the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching.	5-212
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit: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.	5-213
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID:DECimal<y>	Sets the ID of the slow channel enhanced type 12-bit data and 8-bit ID for SENT signal searching in binary notation.	5-213
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:ID: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.	5-213
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit?	Queries all slow channel enhanced type 16-bit data and 4-bit ID settings for SENT signal searching.	5-213
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA?	Queries all slow channel enhanced type 16-bit data and 4-bit ID data settings for SENT signal searching.	5-213
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit: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.	5-213
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA: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.	5-213
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:HEXa	Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in hexadecimal notation.	5-214
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA: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.	5-214
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:PATTern	Sets the slow channel enhanced type 16-bit data and 4-bit ID data for SENT signal searching in binary notation.	5-214
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:PFORmat	Sets or queries the data input format, which is one of the slow channel enhanced type 16-bit data and 4-bit ID data conditions, for SENT signal searching.	5-214
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:ID?	Queries all ID settings related to the slow channel enhanced type 16-bit data and 4-bit ID for SENT signal searching.	5-214
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit: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.	5-214
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:ID: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.	5-215
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:ID: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.	5-215
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt?	Queries all slow channel short type settings for SENT signal searching.	5-215
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA?	Queries all slow channel short type data settings for SENT signal searching.	5-215
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:CONDition	Sets or queries the short type data comparison condition of the SENT signal search slow channel.	5-215
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:DECimal<y>	Sets or queries the SENT signal search slow channel short type data in decimal notation.	5-215

Command	Function	Page
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:HEXa	Sets the SENT signal search slow channel short type data in hexadecimal notation.	5-215
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:MODE	Sets or queries the short type data enable/disable condition of the SENT signal search slow channel.	5-216
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:PATTern	Sets the SENT signal search slow channel short type data in binary notation.	5-216
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:PFORmat	Sets or queries the data input format, which is one of the slow channel short type data conditions, for SENT signal searching.	5-216
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID?	Queries all slow channel short type ID settings for SENT signal searching.	5-216
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:CONDition	Sets or queries the short type ID comparison condition of the SENT signal search slow channel.	5-216
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:DECimal<y>	Sets the SENT signal search slow channel short type ID in decimal notation.	5-216
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:MODE	Sets or queries the short type ID enable/disable condition of the SENT signal search slow channel.	5-216
:SERialbus<x>:SENT:TRENd<y>?	Queries all SENT signal analysis trend display settings.	5-217
:SERialbus<x>:SENT:TRENd<y>:ASCale	Executes auto scaling of the SENT signal analysis trend display.	5-217
:SERialbus<x>:SENT:TRENd<y>:CURSor?	Queries all SENT signal analysis trend display cursor measurement settings.	5-217
:SERialbus<x>:SENT:TRENd<y>:CURSor:DT:VALue?	Queries the time value between cursors on the SENT signal analysis trend display.	5-217
:SERialbus<x>:SENT:TRENd<y>:CURSor:DV:VALue?	Queries the vertical value between cursors on the SENT signal analysis trend display.	5-217
:SERialbus<x>:SENT:TRENd<y>:CURSor:MODE	Sets or queries the automated measurement mode of the SENT signal analysis trend display.	5-217
:SERialbus<x>:SENT:TRENd<y>:CURSor:POSition<z>	Sets or queries the position of the specified cursor on the SENT signal analysis trend display.	5-217
:SERialbus<x>:SENT:TRENd<y>:CURSor:T<z>:VALue?	Queries the time value at the specified cursor on the SENT signal analysis trend display.	5-217
:SERialbus<x>:SENT:TRENd<y>:CURSor:V<z>:VALue?	Queries the vertical value at the specified cursor on the SENT signal analysis trend display.	5-217
:SERialbus<x>:SENT:TRENd<y>:DISPlay	Sets or queries whether to show (ON) or hide (OFF) the SENT signal analysis trend.	5-217
:SERialbus<x>:SENT:TRENd<y>:HRANge	Sets or queries the SENT signal analysis trend display source window.	5-218
:SERialbus<x>:SENT:TRENd<y>:SID	Sets or queries the slow channel ID of the SENT signal analysis trend display in hexadecimal notation.	5-218
:SERialbus<x>:SENT:TRENd<y>:SOURce	Sets or queries the SENT signal analysis trend display source channel.	5-218
:SERialbus<x>:SENT:TRENd<y>:UDATa	Sets or queries the fast channel user-defined data of the SENT signal analysis trend display.	5-218
:SERialbus<x>:SENT:TRENd<y>:VERTical	Sets or queries the vertical range of the SENT signal analysis trend display.	5-218
:SERialbus<x>:SENT:TRENd<y>:VTDisplay	Sets or queries the on/off status of the VT waveform display on the SENT signal analysis trend display.	5-218
:SERialbus<x>:SOURce?	Queries all analysis and search settings.	5-218
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}?	Queries all source waveform settings.	5-218
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:HYSTeresis	Sets or queries a source waveform hysteresis.	5-219
:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:LEVel	Sets or queries a source waveform level.	5-219
:SERialbus<x>:SPATtern?	Queries all user-defined bus signal analysis and search settings.	5-219
:SERialbus<x>:SPATtern:ANALyze?	Queries all user-defined bus signal analysis execution settings.	5-219
:SERialbus<x>:SPATtern[:ANALyze]:SETup?	Queries all user-defined bus signal analysis settings.	5-219
:SERialbus<x>:SPATtern[:ANALyze]:SETup:BRATe	Sets or queries the bit rate for user-defined bus signal analysis.	5-219
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk?	Queries all clock signal settings for user-defined bus signal analysis.	5-219
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:MODE	Sets or queries the clock signal enable or disable status for user-defined bus signal analysis.	5-219

Command	Function	Page
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:POLarity	Sets or queries the clock signal slope for user-defined bus signal analysis.	5-220
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:SOURce	Sets or queries the clock signal for user-defined bus signal analysis.	5-220
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS?	Queries all chip select signal settings for user-defined bus signal analysis.	5-220
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS:ACTive	Sets or queries the chip select signal active state for user-defined bus signal analysis.	5-220
:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS:SOURce	Sets or queries the chip select signal for user-defined bus signal analysis.	5-220
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA?	Queries all data signal settings for user-defined bus signal analysis.	5-220
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:ACTive	Sets or queries the data signal active state for user-defined bus signal analysis.	5-220
:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:SOURce	Sets or queries the data signal for user-defined bus signal analysis.	5-221
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh?	Queries all latch signal settings for user-defined bus signal analysis.	5-221
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:POLarity	Sets or queries the latch signal slope for user-defined bus signal analysis.	5-221
:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:SOURce	Sets or queries the latch signal for user-defined bus signal analysis.	5-221
:SERialbus<x>:SPATtern[:ANALyze]:SETup:SPOint	Sets or queries the analysis start point for user-defined bus signal analysis.	5-221
:SERialbus<x>:SPATtern:SEARch?	Queries all user-defined bus signal search settings.	5-221
:SERialbus<x>:SPATtern:SEARch:ABORt	Aborts the user-defined bus signal search.	5-221
:SERialbus<x>:SPATtern:SEARch:EXECute	Executes a user-defined bus signal search.	5-221
:SERialbus<x>:SPATtern:SEARch: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.	5-222
:SERialbus<x>:SPATtern:SEARch:SELect? Maximum	Queries the number of detected points in the user-defined bus signal search.	5-222
:SERialbus<x>:SPATtern:SEARch:SETup?	Queries all user-defined bus signal search condition settings.	5-222
:SERialbus<x>:SPATtern:SEARch:SETup:BITSize	Sets or queries the bit length setting for user-defined bus signal analysis.	5-222
:SERialbus<x>:SPATtern:SEARch:SETup:HEXa	Sets or queries the data condition for user-defined bus signal searching in hexadecimal notation.	5-222
:SERialbus<x>:SPATtern:SEARch:SETup:PATTern	Sets or queries the data condition for user-defined bus signal searching in binary notation.	5-222
:SERialbus<x>:SPATtern:SEARch:SETup:PFORmat	Sets or queries the input format, which is one of the data conditions, for user-defined bus signal searching.	5-222
:SERialbus<x>:SPI?	Queries all SPI bus signal analysis and search settings.	5-222
:SERialbus<x>:SPI:ANALyze?	Queries all SPI bus signal analysis settings.	5-222
:SERialbus<x>:SPI[:ANALyze]:SETup?	Queries all SPI bus signal analysis bus settings.	5-223
:SERialbus<x>:SPI[:ANALyze]:SETup:BITorder	Sets or queries the bit order of the SPI bus signal analysis data.	5-223
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk?	Queries all clock signal settings for SPI bus signal analysis.	5-223
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:POLarity	Sets or queries the clock signal slope for SPI bus signal analysis.	5-223
:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:SOURce	Sets or queries the clock signal for SPI bus signal analysis.	5-223
:SERialbus<x>:SPI[:ANALyze]:SETup:CS?	Queries all chip select signal settings for SPI bus signal analysis.	5-223
:SERialbus<x>:SPI[:ANALyze]:SETup:CS:ACTive	Sets or queries the chip select signal active state for SPI bus signal analysis.	5-223
:SERialbus<x>:SPI[:ANALyze]:SETup:CS:SOURce	Sets or queries the chip select signal for SPI bus signal analysis.	5-224
:SERialbus<x>:SPI[:ANALyze]:SETup:DATA<y>?	Queries all data signal settings for SPI bus signal analysis.	5-224

Command	Function	Page
:SERialbus<x>:SPI[:ANALyze]:SETup:DATA<y>:SOURce	Sets or queries the data signal for SPI bus signal analysis.	5-224
:SERialbus<x>:SPI[:ANALyze]:SETup:FIELd	Sets or queries the data field size for SPI bus signal analysis.	5-224
SERialbus<x>:SPI[:ANALyze]:SETup:GROuping	Sets or queries the on/off status of grouping for SPI bus signal analysis when there is no chip select signal.	5-224
:SERialbus<x>:SPI[:ANALyze]:SETup:ITIMe	Sets or queries the idle time for SPI bus signal analysis when there is no chip select signal.	5-224
:SERialbus<x>:SPI[:ANALyze]:SETup:MODE	Sets or queries the data signal wiring system (three-wire or four-wire) for SPI bus signal analysis.	5-225
:SERialbus<x>:SPI[:ANALyze]:SETup:MSBLsb	Sets or queries the data MSB and LSB bits for SPI bus signal analysis.	5-225
:SERialbus<x>:SPI:DETail?	Queries all SPI bus signal analysis result list settings.	5-225
:SERialbus<x>:SPI:DETail:DISPlay	Sets or queries the display mode for the SPI bus signal analysis result list.	5-225
:SERialbus<x>:SPI:DETail:LIST:ITEM?	Queries all items that will be displayed in the SPI bus signal analysis result list.	5-225
:SERialbus<x>:SPI:DETail:LIST:VALue?	Queries all of the data for the specified analysis number in the SPI bus signal analysis result list.	5-225
:SERialbus<x>:SPI:SEARch?	Queries all SPI bus signal search settings.	5-225
:SERialbus<x>:SPI:SEARch:ABORt	Aborts the SPI bus signal search.	5-225
:SERialbus<x>:SPI:SEARch:EXECute	Executes a SPI bus signal search.	5-225
:SERialbus<x>:SPI:SEARch:SELect	Sets which detected point to display in the SPI bus signal search zoom window and queries the zoom position of the detected point.	5-225
:SERialbus<x>:SPI:SEARch:SELect? MAXimum	Queries the number of detected points in the SPI bus signal search.	5-226
:SERialbus<x>:SPI:SEARch:SETup?	Queries all SPI bus signal search condition settings.	5-226
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>?	Queries all SPI bus signal search data settings.	5-226
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:BCOunt	Sets or queries the starting position of data comparison for SPI bus signal searching.	5-226
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:CONDition	Sets or queries the data comparison condition (true or false) for SPI bus signal searching.	5-226
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DBYTe	Sets or queries the data size (in bytes) for SPI bus signal searching.	5-226
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:HEXa<z>	Sets the data for SPI bus signal searching in hexadecimal notation.	5-226
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:MODE	Sets or queries the data enable/disable condition for SPI bus signal searching.	5-227
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:PATTern<z>	Sets or queries the data value for SPI bus signal searching in binary notation.	5-227
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:PFORmat	Sets or queries the input format, which is one of the data conditions, for SPI bus signal searching.	5-227
:SERialbus<x>:TYPE	Sets or queries search type.	5-227
:SERialbus<x>:UART?	Queries all UART signal analysis and search settings.	5-227
:SERialbus<x>:UART:ANALyze?	Queries all UART signal analysis settings.	5-227
:SERialbus<x>:UART[:ANALyze]:SETup?	Queries all UART signal analysis bus settings.	5-227
:SERialbus<x>:UART[:ANALyze]:SETup:BITorder	Sets or queries the UART signal analysis bit order.	5-227
:SERialbus<x>:UART[:ANALyze]:SETup:BRATe	Sets or queries the UART signal analysis bit rate (data transfer rate).	5-228
:SERialbus<x>:UART[:ANALyze]:SETup:BSPace	Sets or queries the byte space setting for UART signal analysis.	5-228
:SERialbus<x>:UART[:ANALyze]:SETup:GROuping	Sets or queries the on/off status of grouping for UART signal analysis.	5-228
:SERialbus<x>:UART[:ANALyze]:SETup:POLarity	Sets or queries the polarity setting for UART signal analysis.	5-228
:SERialbus<x>:UART[:ANALyze]:SETup:SOURce	Sets or queries the source signal for UART signal analysis.	5-228
:SERialbus<x>:UART[:ANALyze]:SETup:SPOint	Sets or queries the UART signal analysis sample point.	5-228
:SERialbus<x>:UART:DETail?	Queries all UART signal analysis result list settings.	5-228

Command	Function	Page
:SERialbus<x>:UART:DETail:DISPlay	Sets or queries the display mode for the UART signal analysis result list.	5-229
:SERialbus<x>:UART:DETail:LIST:ITEM?	Queries all items that will be displayed in the UART signal analysis result list.	5-229
:SERialbus<x>:UART:DETail:LIST:VALue?	Queries all of the data for the specified analysis number in the UART signal analysis result list.	5-229
:SERialbus<x>:UART:SEARch?	Queries all UART signal search settings.	5-229
:SERialbus<x>:UART:SEARch:ABORt	Aborts the UART signal search.	5-229
:SERialbus<x>:UART:SEARch:EXECute	Executes a UART signal search.	5-229
:SERialbus<x>:UART:SEARch:SELect	Sets which detected point to display in the UART signal search zoom window and queries the zoom position of the detected point.	5-229
:SERialbus<x>:UART:SEARch:SELect? Maximum	Queries the number of detected points in the UART signal search.	5-229
:SERialbus<x>:UART:SEARch:SETup?	Queries all UART signal search condition settings.	5-229
:SERialbus<x>:UART:SEARch:SETup:DATA?	Queries all UART signal search data settings.	5-229
:SERialbus<x>:UART:SEARch:SETup:DATA:ASCii	Sets the UART bus signal search data in ASCII format.	5-229
:SERialbus<x>:UART:SEARch:SETup:DATA:CONDition	Sets or queries the data comparison condition (true or false) for UART signal searching.	5-230
:SERialbus<x>:UART:SEARch:SETup:DATA:CSENsitive	Sets or queries whether to distinguish uppercase and lowercase letters in ASCII data for the UART bus signal search.	5-230
:SERialbus<x>:UART:SEARch:SETup:DATA:DBYTe	Sets or queries the number of data bytes for UART signal searching.	5-230
:SERialbus<x>:UART:SEARch:SETup:DATA:HEXa<y>	Sets the data for UART signal searching in hexadecimal notation.	5-230
:SERialbus<x>:UART:SEARch:SETup:DATA:PATTern<y>	Sets or queries the data value for UART signal searching in binary notation.	5-230
:SERialbus<x>:UART:SEARch:SETup:DATA:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for UART signal searching.	5-230
:SERialbus<x>:UART:SEARch:SETup:ERRor?	Queries all UART signal search error settings.	5-230
:SERialbus<x>:UART:SEARch:SETup:ERRor:FRAMing	Sets or queries the UART signal search framing error setting.	5-231
:SERialbus<x>:UART:SEARch:SETup:ERRor:PARity	Sets or queries the UART signal search parity error setting.	5-231
:SERialbus<x>:UART:SEARch:SETup:ERRor:PMODe	Sets or queries the UART signal search parity mode setting.	5-231
:SERialbus<x>:UART:SEARch:SETup:FORMat	Sets or queries the format setting for UART signal analysis.	5-231
:SERialbus<x>:UART:SEARch:SETup:MODE	Sets or queries the UART signal search mode.	5-231
:SERialbus<x>:ZLINkage	Sets or queries whether or not the analysis numbers of serial-bus signal-analysis results are linked to zoom locations.	5-231

SNAP Group

:SNAP	Takes a snapshot.	5-232

SSTart Group

:SSTart?	Sets the trigger mode to SINGLE, and starts waveform acquisition. Returns zero if data acquisition is stopped before the specified time period. Returns 1 if waveform acquisition is not stopped within the specified time period.	5-233

STARt Group

:STARt	Starts waveform acquisition.	5-234

STATus Group

:STATus?	Queries all of the settings for the communication status feature.	5-235
:STATus:CONDition?	Queries the contents of the condition register.	5-235
:STATus:EESE	Sets or queries the extended event enable register.	5-235

Command	Function	Page
`:STATus:EESR?`	Queries the contents of the extended event register and clear the register.	5-235
`:STATus:ERRor?`	Queries the error code and message information (top of the error queue).	5-235
`:STATus:FILTer<x>`	Sets or queries the transition filter.	5-235
`:STATus:QENable`	Sets or queries whether or not messages other than errors will be stored to the error queue (on/off).	5-235
`:STATus:QMESsage`	Sets or queries whether or not message information will be attached to the response to the STATus:ERRor? query (on/off).	5-235
`:STATus:SPOLl? (Serial Poll)`	Executes serial polling.	5-235

STOP Group

`:STOP`	Stops waveform acquisition.	5-236

STORe Group

`:STORe?`	Queries all of the information related to setup data in the internal memory.	5-237
`:STORe:SETup<x>?`	Queries information about the setup data in the specified location of the internal memory.	5-237
`:STORe:SETup<x>:COMMent`	Sets or queries the comment for the setup data that is stored to the specified location in the internal memory.	5-237
`:STORe:SETup<x>:DATE?`	Queries the date and time of the setup data that is stored to the specified location in the internal memory.	5-237
`:STORe:SETup<x>:EXECute`	Saves setup data to the specified location in the internal memory.	5-237
`:STORe:SETup<x>:LOCK`	Sets or queries the data-write protection on/off status for the setup data that is stored to the specified location in the internal memory.	5-237

SYSTem Group

`:SYSTem?`	Queries all system settings.	5-238
`:SYSTem:BEEP`	Generates a beep sound.	5-238
`:SYSTem:CLICk`	Sets or queries the click sound on/off status.	5-238
`:SYSTem:CLOCk?`	Queries all date/time settings.	5-238
`:SYSTem:CLOCk:DATE`	Sets or queries the date.	5-238
`:SYSTem:CLOCk:FORMat`	Sets or queries the date format.	5-238
`:SYSTem:CLOCk:MODE`	Sets or queries the on/off status of the date/time display.	5-238
`:SYSTem:CLOCk:SNTP?`	Queries the date/time setting retrieved using SNTP.	5-238
`:SYSTem:CLOCk:SNTP:EXECute`	Sets the date and time using SNTP.	5-238
`:SYSTem:CLOCk:SNTP:GMTTime`	Sets or queries the time difference from Greenwich Mean Time.	5-238
`:SYSTem:CLOCk:TIME`	Sets or queries the time.	5-238
`:SYSTem:DCANcel (Delay Cancel)`	Sets or queries whether or not the specified delay value will be applied to time measurement (on/off).	5-238
`:SYSTem:FSIZe? (Font Size)`	Queries all font size settings.	5-238
`:SYSTem:FSIZe:MEASure`	Sets or queries the font size that is used to display the automatically measured waveform parameters and the cursor measurement values.	5-238
`:SYSTem:LANGuage`	Sets or queries the message language.	5-239
`:SYSTem:LCD?`	Queries all LCD settings.	5-239
`:SYSTem:LCD:AUTO?`	Queries all of the settings for the feature that automatically turns off the LCD backlight.	5-239
`:SYSTem:LCD:AUTO:MODE`	Sets or queries the on/off status of the feature that automatically turns off the LCD backlight.	5-239
`:SYSTem:LCD:AUTO:TIME`	Sets or queries the amount of time until the LCD backlight is turned off.	5-239
`:SYSTem:LCD:BRIGhtness`	Sets or queries the LCD brightness.	5-239
`:SYSTem:LCD:MODE`	Sets or queries the on/off status of the LCD backlight.	5-239
`:SYSTem:LMODe (Legacy Mode)`	Sets or queries whether to initialize the settings to the default values of legacy models when Default Setup is executed.	5-239
`:SYSTem:MLANguage`	Sets or queries the menu language.	5-239
`:SYSTem:OCANcel (Offset Cancel)`	Sets or queries whether or not the specified offset voltage will be applied to measurement and computation (on/off).	5-239
`:SYSTem:TOUT?`	Queries all trigger out settings.	5-239
`:SYSTem:TOUT:POLarity`	Sets or queries the trigger out polarity.	5-239
`:SYSTem:USBKeyboard`	Sets or queries the USB keyboard type.	5-239

TIMebase Group

`:TIMebase?`	Queries all time base settings.	5-240

Command	Function	Page
:TIMebase:SRATe? (Sample RATE)	Queries the sample rate.	5-240
:TIMebase:TDIV	Sets or queries the Time/div value.	5-240

TRIGger Group

:TRIGger?	Queries all trigger settings.	5-241
:TRIGger:ABN?	Queries all A->B(n) trigger settings.	5-241
:TRIGger:ABN:COUNt	Sets or queries the number of times condition B must be met for the A->B(n) trigger.	5-241
:TRIGger:ACTion?	Queries all action-on-trigger settings.	5-241
:TRIGger:ACTion:ACQCount	Sets or queries the action-on-trigger action.	5-241
:TRIGger:ACTion:BUZZer	Sets or queries the whether or not the DLM2000 will sound an alarm as an action when trigger conditions are met (on/off).	5-241
:TRIGger:ACTion:HCOPy	Sets or queries whether or not the DLM2000 will print a screen capture as an action when trigger conditions are met (on/off).	5-242
:TRIGger:ACTion:MAIL?	Queries all of the settings for email notification that is sent when trigger conditions are met.	5-242
:TRIGger:ACTion:MAIL:COUNt	Sets or queries the upper limit of email notifications that are sent when trigger conditions are met.	5-242
:TRIGger:ACTion:MAIL:MODE	Sets or queries whether or not the DLM2000 will send email notification as an action.	5-242
:TRIGger:ACTion:SAVE Function	Sets or queries whether or not the DLM2000 will save waveform data to the storage medium as an action when trigger conditions are met (on/off).	5-242
:TRIGger:ACTion:STARt	Starts the action-on-trigger operation.	5-242
:TRIGger:ACTion:STOP	Stops the action-on-trigger operation.	5-242
:TRIGger:ADB?	Queries all A Delay B trigger settings.	5-242
:TRIGger:ADB:DELay	Sets or queries the delay time for condition B for the A Delay B trigger.	5-242
:TRIGger{[:ATRigger]|:BTRigger}?	Queries all trigger condition settings.	5-242
:TRIGger{[:ATRigger]|:BTRigger}:CAN?	Queries all CAN bus signal trigger settings.	5-242
:TRIGger{[:ATRigger]|:BTRigger}:CAN:BRATe	Sets or queries the CAN bus signal trigger bit rate (data transfer rate).	5-242
:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame?	Queries all CAN bus signal trigger error settings.	5-242
:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame:CRC	Sets or queries the CAN bus signal trigger CRC error setting.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame[:MODE]	Sets or queries the CAN bus signal trigger error frame setting.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame:STUFF	Sets or queries the CAN bus signal trigger stuff error setting.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]?	Queries all ID and data condition settings for CAN bus signal triggering.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK?	Queries all ACK settings for the ID and data conditions for CAN bus signal triggering.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK:MODE	Sets or queries the ACK mode, which is one of the ID and data conditions, for CAN bus signal triggering.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:ACK:TYPE	Sets or queries the ACK condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA?	Queries all data settings for the ID and data conditions for CAN bus signal triggering.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:CONDition	Sets or queries the comparison condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-243
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DECimal<x>	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal triggering in decimal notation.	5-244
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DLC	Sets or queries the data length code (DLC), which is one of the ID and data conditions, for CAN bus signal triggering.	5-244
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA: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.	5-244
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:HEXa<x>	Sets a reference value, which is one of the ID and data conditions, for CAN bus signal triggering in hexadecimal notation.	5-244
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MODE	Sets or queries the data enable/disable condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-244

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA: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.	5-244
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PATTern<x>	Sets or queries the data value, which is one of the ID and data conditions, for CAN bus signal triggering in binary notation.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for CAN bus signal triggering.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:SIGN	Sets or queries whether signed or unsigned data format will be used, which is one of the ID and data conditions, for CAN bus signal triggering.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier?	Queries all identifier settings for the ID and data conditions for CAN bus signal triggering.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID?	Queries all ID settings for the ID and data conditions for CAN bus signal triggering.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:HEXa	Sets the ID value, which is one of the ID and data conditions, for CAN bus signal triggering in hexadecimal notation.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:MODE	Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:PATTern	Sets or queries the ID pattern, which is one of the ID and data conditions, for CAN bus signal triggering in binary notation.	5-245
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier: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.	5-246
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for CAN bus signal triggering.	5-246
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal?	Queries all message and signal settings for the ID and data conditions for CAN bus signal triggering.	5-246
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:MESSage:ITEM	Sets the message item, which is one of the ID and data conditions, for CAN bus signal triggering.	5-246
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SELect	Sets or queries the message and signal conditions for CAN bus signal triggering.	5-246
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal?	Queries all signal settings for the ID and data conditions for CAN bus signal triggering.	5-246
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:CONDition	Sets or queries the signal data condition, which is one of the ID and data conditions, for CAN bus signal triggering.	5-246
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:DECimal<x>	Sets a signal reference value, which is one of the ID and data conditions, for CAN bus signal triggering in decimal notation.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:ITEM	Sets the signal item, which is one of the ID and data conditions, for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:RTR	Sets or queries the RTR value, which is one of the ID and data conditions, for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR?	Queries all IDOR condition settings for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK?	Queries all IDOR condition ACK settings for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:MODE	Sets or queries the ACK mode, which is one of the IDOR conditions, for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:TYPE	Sets or queries the ACK condition, which is one of the IDOR conditions, for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA?	Queries all data settings for the IDOR condition for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA[:MODE]	Sets or queries the data enable/disable condition, which is one of the IDOR conditions, for CAN bus signal triggering.	5-247
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier?	Queries all identifier settings for the IDOR condition for CAN bus signal triggering.	5-248
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>?	Queries all IDOR condition ID settings for CAN bus signal triggering.	5-248
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:HEXa	Sets the ID value, which is one of the IDOR conditions, for CAN bus signal triggering in hexadecimal notation.	5-248

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:MODE	Sets or queries the ID enable/disable condition, which is one of the IDOR conditions, for CAN bus signal triggering.	5-248
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:PATTern	Sets or queries the ID pattern, which is one of the IDOR conditions, for CAN bus signal triggering in binary notation.	5-248
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MODE	Sets or queries the identifier enable/disable condition, which is one of the IDOR conditions, for CAN bus signal triggering.	5-248
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the IDOR conditions, for CAN bus signal triggering.	5-248
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the IDOR conditions, for CAN bus signal triggering.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:MSIGnal<x>:MESSage:ITEM	Sets the message item, which is one of the IDOR conditions, for CAN bus signal triggering.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:RTR	Sets or queries the RTR value, which is one of the IDOR conditions, for CAN bus signal triggering.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:CAN:MODE	Sets or queries the CAN bus signal trigger mode setting.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:CAN:RECessive	Sets or queries the CAN bus signal trigger recessive level.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:CAN:SOURce	Sets or queries the CAN bus signal trigger source signal.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:CAN:SPOint	Sets or queries the CAN bus signal trigger sample point.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray?	Queries all FLEXRAY bus signal trigger settings.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BCHannel	Sets or queries the channel bus type for FLEXRAY bus signal triggering.	5-249
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BRATe	Sets or queries the FLEXRAY bus signal trigger bit rate (data transfer rate).	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor?	Queries all FLEXRAY bus signal trigger error settings.	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:BSS	Sets or queries the FLEXRAY bus signal trigger BSS error setting.	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:CRC	Sets or queries the FLEXRAY bus signal trigger CRC error setting.	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:ERRor:FES	Sets or queries the FLEXRAY bus signal trigger FES error setting.	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDData?	Queries all ID and data condition settings for FLEXRAY bus signal triggering.	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt?	Queries all cycle-count settings for the ID and data conditions for FLEXRAY bus signal triggering.	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition	Sets or queries the cycle-count data condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-250
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt<x>	Sets or queries the cycle count, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-251
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:MODE	Sets or queries the cycle-count enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-251
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>?	Queries all data field settings for the ID and data conditions for FLEXRAY bus signal triggering.	5-251
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:BCOunt	Sets or queries the position for comparing data field data patterns, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-251
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:CONDition	Sets or queries the data field data condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-251
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DBYTe	Sets or queries the number of data field data bytes, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-252
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DECimal<y>	Sets a reference value, which is one of the ID and data conditions, for FLEXRAY bus signal triggering in hexadecimal notation.	5-252

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:ENDian	Sets or queries the data field endian setting, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-252
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:HEXa<y>	Sets a data field reference value, which is one of the ID and data conditions, for FLEXRAY bus signal triggering in hexadecimal notation.	5-252
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MODE	Sets or queries the data field enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-252
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MSBLsb	Sets or queries the MSB and LSB bits for the data field, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-252
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:PATTern<y>	Sets or queries the data field data, which is one of the ID and data conditions, for FLEXRAY bus signal triggering in binary notation.	5-253
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:PFORmat	Sets or queries the data field data input format, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-253
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:SIGN	Sets or queries whether a signed or unsigned data format will be used for the data field (this is one of the ID and data conditions) for FLEXRAY bus signal triggering.	5-253
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID?	Queries all frame ID settings for the ID and data conditions for FLEXRAY bus signal triggering.	5-253
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:CONDition	Sets or queries the frame ID data condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-253
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:ID<x>	Sets or queries the Frame ID value, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-253
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:MODE	Sets or queries the Frame ID enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-253
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator?	Queries all indicator settings for the ID and data conditions for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:MODE	Sets or queries the indicator enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:NFRame	Sets or queries the indicator null frame, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:PPReamble	Sets or queries the indicator payload preamble, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:STFRame	Sets or queries the indicator start frame, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:SYFRame	Sets or queries the indicator sync frame, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR?	Queries all OR condition settings for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>?	Queries all the settings of the frame IDs, which are OR conditions, for FLEXRAY bus signal triggering.	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt?	Queries the cycle-count setting for the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).	5-254
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:CONDition	Sets or queries the comparison condition for the cycle counts of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).	5-255
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:COUNt<y>	Sets or queries the reference value for the cycle counts of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).	5-255
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID?	Queries all the ID conditions of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).	5-255
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:CONDition	Sets or queries the ID comparison condition of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).	5-255
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:ID<y>	Sets or queries the ID reference value for the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).	5-255

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:MODE	Sets or queries the frame ID enable/disable condition, which is one of the OR conditions, for FLEXRAY bus signal triggering.	5-255
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:MODE	Sets or queries the FLEXRAY bus signal trigger mode.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:SOURce	Sets or queries the trigger source for FLEXRAY bus signal triggering.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C?	Queries all I^{2}C bus signal trigger settings.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess?	Queries all address pattern settings for I^{2}C bus signal triggering.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess?	Queries all address pattern address settings for I^{2}C bus signal triggering.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress?	Queries all 10-bit address settings for I^{2}C bus signal triggering.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:HEXa	Sets the 10-bit address for I^{2}C bus signal triggering in hexadecimal notation.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:PATTern	Sets or queries the 10-bit address for I^{2}C bus signal triggering in binary notation.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]?	Queries all 7-bit address settings for I^{2}C bus signal triggering.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:HEXa	Sets the 7-bit address for I^{2}C bus signal triggering in hexadecimal notation.	5-256
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:PATTern	Sets or queries the 7-bit address for I^{2}C bus signal triggering in binary notation.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub?	Queries all 7-bit + sub address settings for I^{2}C bus signal triggering.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess?	Queries all 7-bit address settings for the 7-bit + sub address for I^{2}C bus signal triggering.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:HEXa	Sets the 7-bit address for the 7-bit + sub address for I^{2}C bus signal triggering in hexadecimal notation.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:PATTern	Sets or queries the 7-bit address for the 7-bit + sub address for I^{2}C bus signal triggering in binary notation.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress?	Queries all sub address settings for the 7-bit + sub address for I^{2}C bus signal triggering.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:HEXa	Sets the sub address for the 7-bit + sub address for I^{2}C bus signal triggering in hexadecimal notation.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:PATTern	Sets or queries the sub address for the 7-bit + sub address for I^{2}C bus signal triggering in binary notation.	5-257
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:MODE	Sets or queries the address condition enable/disable mode for I^{2}C bus signal triggering.	5-258
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:PFORmat	Sets or queries the address input format, which is one of the address conditions, for I^{2}C bus signal triggering.	5-258
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:TYPE	Sets or queries the address type, which is one of the address conditions, for I^{2}C bus signal triggering.	5-258
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA?	Queries all address pattern data settings for I^{2}C bus signal triggering.	5-258
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BCOunt	Sets or queries the position for comparing data patterns for I^{2}C bus signal triggering.	5-258
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BMODe	Sets or queries the on/off status of the position for comparing data patterns for I^{2}C bus signal triggering.	5-258
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:CONDition	Sets or queries the data comparison condition (true or false) for I^{2}C bus signal triggering.	5-258

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:DBYTe	Sets or queries the specified number of data bytes for I^2C bus signal triggering.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:HEXa<x>	Sets the data for I^2C bus signal triggering in hexadecimal notation.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:MODE	Sets or queries the data condition enable/disable mode for I^2C bus signal triggering.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:PATTern<x>	Sets or queries the data value, which is one of the data conditions, for I^2C bus signal triggering in binary notation.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:PFORmat	Sets or queries the data input format, which is one of the data conditions, for I^2C bus signal triggering.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall?	Queries all general call settings for I^2C bus signal triggering.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress?	Queries all general call's 7-bit master address settings for I^2C bus signal triggering.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:HEXa	Sets the general call's 7-bit master address for I^2C bus signal triggering in hexadecimal notation.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:PATTern	Sets or queries the general call's 7-bit master address for I^2C bus signal triggering in binary notation.	5-259
:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:SBYTe	Sets or queries the general call's second byte type for I^2C bus signal triggering.	5-260
:TRIGger{[:ATRigger]|:BTRigger}:I2C:INCLuderw	Sets or queries the on/off status of the R/W address bit in I^2C bus signal triggering.	5-260
:TRIGger{[:ATRigger]|:BTRigger}:I2C:MODE	Sets or queries the trigger type for I^2C bus signal triggering.	5-260
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack?	Queries all NON ACK ignore mode settings for I^2C bus signal triggering.	5-260
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:HSMode	Sets or queries whether or not NON ACK will be ignored in high-speed mode for I^2C bus signal triggering.	5-260
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:READaccess	Sets or queries whether or not NON ACK will be ignored in read access mode for I^2C bus signal triggering.	5-260
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:STARtbyte	Sets or queries whether or not NON ACK will be ignored in start bytes for I^2C bus signal triggering.	5-260
:TRIGger{[:ATRigger]|:BTRigger}:I2C:SCL	Sets or queries the clock signal for I^2C bus signal triggering.	5-261
:TRIGger{[:ATRigger]|:BTRigger}:I2C:SDA	Sets or queries the data signal for I^2C bus signal triggering.	5-261
:TRIGger{[:ATRigger]|:BTRigger}:LIN?	Queries all LIN bus signal trigger settings.	5-261
:TRIGger{[:ATRigger]|:BTRigger}:LIN:BLENgth	Sets or queries the LIN bus signal trigger break length.	5-261
:TRIGger{[:ATRigger]|:BTRigger}:LIN:BRATe	Sets or queries the LIN bus signal trigger bit rate (data transfer rate).	5-261
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor?	Queries all LIN bus signal trigger error settings.	5-261
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:PARity	Sets or queries the LIN bus signal trigger parity error setting.	5-261
:TRIGger{[:ATRigger]|:BTRigger}:LIN:ERRor:SYNCh	Sets or queries the LIN bus signal trigger synch error setting.	5-261
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData?	Queries all ID and data settings for LIN bus signal triggering.	5-262
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA?	Queries all LIN bus signal trigger data settings.	5-262
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:CONDition	Sets or queries the data conditions for LIN bus signal triggering.	5-262
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DBYTe	Sets or queries the number of data bytes for LIN bus signal triggering.	5-262
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:DECimal<x>	Sets the data for LIN bus signal triggering in decimal notation.	5-262
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:ENDian	Sets or queries the data endian setting for LIN bus signal triggering.	5-262
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:HEXa<x>	Sets the data for LIN bus signal triggering in hexadecimal notation.	5-262

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MODE	Sets or queries the data enable/disable condition for LIN bus signal triggering.	5-262
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MSBLsb	Sets or queries the data MSB and LSB bits for LIN bus signal triggering.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:PATTern<x>	Sets or queries the data value for LIN bus signal triggering in binary notation.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:PFORmat	Sets or queries the input format, which is one of the data conditions, for LIN bus signal triggering.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:SIGN	Sets or queries whether signed or unsigned data format will be used, which is one of the data conditions, for LIN bus signal triggering.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier?	Queries all LIN bus signal trigger identifier settings.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID?	Queries all LIN bus signal trigger ID settings.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:HEXa	Sets the ID for LIN bus signal triggering in hexadecimal notation.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:MODE	Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for LIN bus signal triggering.	5-263
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:PATTern	Sets or queries the ID value for LIN bus signal triggering in binary notation.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the ID and data conditions, for LIN bus signal triggering.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR?	Queries all IDOR condition settings for LIN bus signal triggering.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier?	Queries all IDOR condition ID settings for LIN bus signal triggering.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>?	Queries an ID value, which is one of the IDOR condition settings, for LIN bus signal triggering.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:HEXa	Sets the ID value, which is one of the IDOR conditions, for LIN bus signal triggering in hexadecimal notation.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:MODE	Sets or queries the ID enable/disable condition, which is one of the IDOR conditions, for LIN bus signal triggering.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:PATTern	Sets or queries the ID pattern, which is one of the IDOR conditions, for LIN bus signal triggering in binary notation.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:MODE	Sets or queries the ID enable/disable condition for LIN bus signal triggering.	5-264
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the IDOR conditions, for LIN bus signal triggering.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:LIN:MODE	Sets or queries the LIN bus signal trigger mode setting.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:LIN:SOURce	Sets or queries the LIN bus signal trigger source signal.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:LIN:SPOint	Sets or queries the LIN bus signal trigger sample point.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:PATTern?	Queries all state trigger settings.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS?	Queries all logic input settings for state triggering.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:HEXa	Sets the logic input state for state triggering in hexadecimal notation.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:PATTern	Sets or queries the logic input state pattern for state triggering.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CHANnel<x>	Sets or queries the state of a channel for state triggering.	5-265
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk?	Queries all clock signal settings for state triggering.	5-266
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SLOPe	Sets or queries the clock signal slope for state triggering.	5-266
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SOURce	Sets or queries the clock signal for state triggering.	5-266
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CONDition	Sets or queries the trigger condition for state triggering.	5-266

5.1 List of Commands

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger} :PATTern:LOGic	Sets or queries the combination condition for state triggering.	5-266
:TRIGger{[:ATRigger]|:BTRigger} :QUALify?	Queries all edge qualify trigger settings.	5-266
:TRIGger{[:ATRigger]|:BTRigger} :QUALify:CHANnel<x>	Sets or queries the conditions for a waveform for edge qualified triggering.	5-266
:TRIGger{[:ATRigger]|:BTRigger} :QUALify:CONDition	Sets or queries the trigger source qualify conditions for edge qualified triggering.	5-266
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle?	Queries all edge trigger settings.	5-266
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:COUPling	Sets or queries the edge trigger source trigger coupling.	5-267
:TRIGger{[:ATRigger]|:BTR igger}:SIMPle:HFRejection (HighFrequencyREJECTION)	Sets or queries the on/off status of the trigger source low-pass filter (HF rejection) for edge triggering.	5-267
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:HYSTeresis	Sets or queries the noise rejection setting for the edge trigger source trigger level.	5-267
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:LEVel	Sets or queries the edge trigger source trigger level.	5-267
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:PROBe	Sets or queries the external trigger source probe setting for edge triggering.	5-267
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:RANGe	Sets or queries the external trigger source input range for edge triggering.	5-267
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:SLOPe	Sets or queries trigger source trigger slope (the polarity when window is set to ON) for edge triggering.	5-268
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:SOURce	Sets or queries the edge trigger source.	5-268
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:WIDTh	Sets or queries the edge trigger source window width.	5-268
:TRIGger{[:ATRigger]|:BTRigger} :SIMPle:WINDow	Sets or queries the edge trigger source window.	5-268
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern?	Queries all user-defined bus signal trigger settings.	5-268
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:BITSize	Sets or queries the bit length setting for user-defined bus signal triggering.	5-268
TRIGger{[:ATRigger]|:BTRigger}: SPATtern:BRATe	Sets or queries the bit rate setting for user-defined bus signal triggering.	5-268
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:CLOCk?	Queries all clock signal settings for user-defined bus signal triggering.	5-268
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:CLOCk:MODE	Sets or queries the clock signal enable or disable status for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:CLOCk:POLarity	Sets or queries the clock signal polarity for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:CLOCk:SOURce	Sets or queries the clock signal for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:CS?	Queries all chip select signal settings for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:CS:ACTive	Sets or queries the chip select signal active state for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:CS:SOURce	Sets or queries the chip select signal for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:DATA?	Queries all data signal settings for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:DATA:ACTive	Sets or queries the data signal active state for user-defined bus signal triggering.	5-269
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:DATA:SOURce	Sets or queries the data signal for user-defined bus signal triggering.	5-270
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:HEXa	Sets or queries the data condition for user-defined bus signal triggering in hexadecimal notation.	5-270
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:LATCh?	Queries all latch signal settings for user-defined bus signal triggering.	5-270
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:LATCh:POLarity	Sets or queries the latch signal polarity for user-defined bus signal triggering.	5-270

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:LATCh:SOURce	Sets or queries the latch signal for user-defined bus signal triggering.	5-270
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:PATTern	Sets or queries the data condition for user-defined bus signal triggering in binary notation.	5-270
:TRIGger{[:ATRigger]|:BTRigger} :SPATtern:PFORmat	Sets or queries the input format, which is one of the data conditions, for user-defined bus signal triggering.	5-270
:TRIGger{[:ATRigger]|:BTRigger} :SPI?	Queries all SPI bus signal trigger settings.	5-270
:TRIGger{[:ATRigger]|:BTRigger} :SPI:BITorder	Sets or queries the bit order of the SPI bus signal trigger data.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:CLOCk?	Queries all clock signal settings for SPI bus signal triggering.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:CLOCk:POLarity	Sets or queries the clock signal polarity for SPI bus signal triggering.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:CLOCk:SOURce	Sets or queries the clock signal for SPI bus signal triggering.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:CS?	Queries all chip select signal settings for SPI bus signal triggering.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:CS:ACTive	Sets or queries the chip select signal active state for SPI bus signal triggering.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:CS:SOURce	Sets or queries the chip select signal for SPI bus signal triggering.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>?	Queries all data signal settings for SPI bus signal triggering.	5-271
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:BCOunt	Sets or queries the starting position of data comparison for SPI bus signal triggering.	5-272
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:CONDition	Sets or queries the data comparison condition (true or false) for SPI bus signal triggering.	5-272
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:DBYTe	Sets or queries the data size (in bytes) for SPI bus signal triggering.	5-272
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:HEXa<y>	Sets a data value for SPI bus signal triggering in hexadecimal notation.	5-272
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:MODE	Sets or queries the data enable/disable condition for SPI bus signal triggering (on/off).	5-272
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:PATTern<y>	Sets or queries the data value for SPI bus signal triggering in binary notation.	5-272
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:PFORmat	Sets or queries the data input format, which is one of the data conditions, for SPI bus signal triggering.	5-272
:TRIGger{[:ATRigger]|:BTRigger} :SPI:DATA<x>:SOURce	Sets or queries the data signal for SPI bus signal triggering.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :SPI:MODE	Sets or queries the wiring system (three-wire or four-wire) for SPI bus signal triggering.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :TYPE	Sets or queries the trigger type.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :UART?	Queries all UART signal trigger settings.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :UART:BITorder	Sets or queries the UART signal trigger bit order.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :UART:BRATe	Sets or queries the UART signal trigger bit rate (data transfer rate).	5-273
:TRIGger{[:ATRigger]|:BTRigger} :UART:DATA?	Queries all UART signal trigger data settings.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :UART:DATA:ASCii	Sets the UART bus signal trigger data in ASCII format.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :UART:DATA:CONDition	Sets or queries the data comparison condition (true or false) for UART signal triggering.	5-273
:TRIGger{[:ATRigger]|:BTRigger} :UART:DATA:CSENsitive	Sets or queries whether to distinguish uppercase and lowercase letters in ASCII data for the UART bus signal trigger.	5-274
:TRIGger{[:ATRigger]|:BTRigger} :UART:DATA:DBYTe	Sets or queries the number of data bytes for UART signal triggering.	5-274
:TRIGger{[:ATRigger]|:BTRigger} :UART:DATA:HEXa<x>	Sets the data for UART signal triggering in hexadecimal notation.	5-274

Command	Function	Page
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:PATTern<x>	Sets or queries the data value for UART signal triggering in binary notation.	5-274
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:PFORmat	Sets or queries the data input format, which is one of the ID and data conditions, for UART signal triggering.	5-274
:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor?	Queries all UART signal trigger error settings.	5-274
:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor:FRAMing	Sets or queries the UART signal trigger framing error setting.	5-274
:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor:PARity	Sets or queries the UART signal trigger parity error setting.	5-275
:TRIGger{[:ATRigger]|:BTRigger}:UART:ERRor:PMODE	Sets or queries the UART signal trigger parity mode setting.	5-275
:TRIGger{[:ATRigger]|:BTRigger}:UART:FORMat	Sets or queries the UART signal trigger format.	5-275
:TRIGger{[:ATRigger]|:BTRigger}:UART:MODE	Sets or queries the UART signal trigger type.	5-275
:TRIGger{[:ATRigger]|:BTRigger}:UART:POLarity	Sets or queries the UART signal trigger polarity.	5-275
:TRIGger{[:ATRigger]|:BTRigger}:UART:SOURce	Sets or queries the source signal for UART signal triggering.	5-275
:TRIGger{[:ATRigger]|:BTRigger}:UART:SPOint	Sets or queries the UART signal trigger sample point.	5-275
:TRIGger[:ATRigger]:CANFD?	Queries all CAN FD bus signal trigger settings.	5-275
:TRIGger[:ATRigger]:CANFD:BRATe	Sets or queries the CAN FD bus signal trigger bit rate (data transfer rate).	5-275
:TRIGger[:ATRigger]:CANFD:DBRate	Sets or queries the CAN FD bus signal trigger data phase bit rate (data transfer rate).	5-276
:TRIGger[:ATRigger]:CANFD:DSPoint	Sets or queries the CAN FD bus signal trigger data phase sample point.	5-276
:TRIGger[:ATRigger]:CANFD:ID?	Queries all ID and data condition settings for CAN FD bus signal triggering.	5-276
:TRIGger[:ATRigger]:CANFD[:ID]:DATA:MODE	Sets or queries the data enable/disable condition, which is one of the ID conditions, for CAN FD bus signal triggering.	5-276
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier?	Queries all identifier settings for the ID conditions for CAN FD bus signal triggering.	5-276
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID?	Queries all ID settings for the ID conditions for CAN FD bus signal triggering.	5-276
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:HEXa	Sets the ID value, which is one of the ID conditions, for CAN FD bus signal triggering in hexadecimal notation.	5-276
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:MODE	Sets or queries the ID enable/disable condition, which is one of the ID conditions, for CAN FD bus signal triggering.	5-276
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:PATTern	Sets or queries the ID condition, which is one of the ID conditions, for CAN FD bus signal triggering in binary notation.	5-276
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the ID conditions, for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the ID conditions, for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD[:ID]:MSIGnal:MESSage:ITEM	Sets the message item, which is one of the ID conditions, for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD[:ID]:RTR	Sets or queries the RTR, which is one of the ID conditions, for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD:IDOR?	Queries all IDOR condition settings for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD:IDOR:DATA[:MODE]	Sets or queries the data enable/disable condition, which is one of the IDOR conditions, for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier?	Queries all identifier settings for the IDOR conditions for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>?	Queries all ID settings for the IDOR conditions for CAN FD bus signal triggering.	5-277
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:HEXa	Sets the ID value, which is one of the IDOR conditions, for CAN FD bus signal triggering in hexadecimal notation.	5-277
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:MODE	Sets or queries the ID enable/disable condition, which is one of the IDOR conditions, for CAN FD bus signal triggering.	5-278
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:PATTern	Sets or queries the ID condition, which is one of the IDOR conditions, for CAN FD bus signal triggering in binary notation.	5-278

Command	Function	Page
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:MODE	Sets or queries the identifier enable/disable condition, which is one of the IDOR conditions, for CAN FD bus signal triggering.	5-278
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:MFORmat	Sets or queries the ID frame format (standard or extended), which is one of the IDOR conditions, for CAN FD bus signal triggering.	5-278
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:PFORmat	Sets or queries the ID input format, which is one of the IDOR conditions, for CAN FD bus signal triggering.	5-278
:TRIGger[:ATRigger]:CANFD:IDOR:MSIGnal<x>:MESSage:ITEM	Sets the message item, which is one of the IDOR conditions, for CAN FD bus signal triggering.	5-278
:TRIGger[:ATRigger]:CANFD:IDOR:RTR	Sets or queries the RTR, which is one of the IDOR conditions, for CAN FD bus signal triggering.	5-278
:TRIGger[:ATRigger]:CANFD:MODE	Sets or queries the CAN FD bus signal trigger mode.	5-279
:TRIGger[:ATRigger]:CANFD:RECessive	Sets or queries the CAN FD bus signal trigger recessive level.	5-279
:TRIGger[:ATRigger]:CANFD:SOURce	Sets or queries the CAN FD bus signal trigger signal.	5-279
:TRIGger[:ATRigger]:CANFD:SPOint	Sets or queries the CAN FD bus signal trigger sample point.	5-279
:TRIGger[:ATRigger]:OR?	Queries all edge OR trigger settings.	5-279
:TRIGger[:ATRigger]:OR:CHANnel<x>	Sets or queries the edge of a channel for edge OR triggering.	5-279
:TRIGger[:ATRigger]:PSI5?	Queries all PSI5 signal trigger settings.	5-279
:TRIGger[:ATRigger]:PSI5:DATA?	Queries all PSI5 signal trigger data settings.	5-279
:TRIGger[:ATRigger]:PSI5:DATA:CONDition	Sets or queries the data condition for PSI5 signal triggering.	5-279
:TRIGger[:ATRigger]:PSI5:DATA:DECimal	Sets or queries the data for PSI5 signal triggering in decimal notation.	5-279
:TRIGger[:ATRigger]:PSI5:DATA:HEXa	Sets the data for PSI5 signal triggering in hexadecimal notation.	5-280
:TRIGger[:ATRigger]:PSI5:DATA:PATTern	Sets or queries the data for PSI5 signal triggering in binary notation.	5-280
:TRIGger[:ATRigger]:PSI5:DATA:PFORmat	Sets or queries the data input format for PSI5 signal triggering.	5-280
:TRIGger[:ATRigger]:PSI5:MODE	Sets or queries the PSI5 signal trigger mode.	5-280
:TRIGger[:ATRigger]:PSI5:SETup?	Queries all bus setup settings for PSI5 signal triggering.	5-280
:TRIGger[:ATRigger]:PSI5:SETup:DATA?	Queries all PSI5 signal trigger data signal settings.	5-280
:TRIGger[:ATRigger]:PSI5:SETup:DATA:BRATe	Sets or queries the PSI5 signal trigger bit rate (data transfer rate).	5-280
:TRIGger[:ATRigger]:PSI5:SETup:DATA:DBITs	Sets or queries the data length for PSI5 signal triggering.	5-281
:TRIGger[:ATRigger]:PSI5:SETup:DATA:EDETection	Sets or queries the error detection method for PSI5 signal triggering.	5-281
:TRIGger[:ATRigger]:PSI5:SETup:DATA:SOURce	Sets or queries the data source for PSI5 signal triggering.	5-281
:TRIGger[:ATRigger]:PSI5:SETup:SYNC	Sets or queries the sync signal for PSI5 signal triggering.	5-281
:TRIGger[:ATRigger]:PULSe?	Queries all pulse width trigger settings.	5-281
:TRIGger[:ATRigger]:PULSe:POLarity	Sets or queries the pulse width trigger source polarity.	5-281
:TRIGger[:ATRigger]:PULSe:SOURce	Sets or queries the pulse width trigger source.	5-281
:TRIGger[:ATRigger]:SENT?	Queries all SENT signal trigger settings.	5-281
:TRIGger[:ATRigger]:SENT:CTICk	Sets or queries the SENT signal trigger clock tick value.	5-281
:TRIGger[:ATRigger]:SENT:FLENgth	Sets or queries the SENT signal trigger frame length (clock ticks).	5-281
:TRIGger[:ATRigger]:SENT:PPULse	Sets or queries the presence or absence of pause pulses for SENT signal triggering.	5-282
:TRIGger[:ATRigger]:SENT:SOURce	Sets or queries the SENT signal trigger source.	5-282
:TRIGger[:ATRigger]:SENT:VERSion	Sets or queries the SENT signal analysis specification version.	5-282
:TRIGger[:ATRigger]:TV?	Queries all TV trigger settings.	5-282
:TRIGger[:ATRigger]:TV:FIELd	Sets or queries the field where TV triggering will be used.	5-282
:TRIGger[:ATRigger]:TV:FRAMe	Sets or queries the frame skip feature for TV triggering.	5-282

Command	Function	Page
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PAL|SDTV|USERdefine}?	Queries all TV trigger mode settings.	5-282
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PAL|SDTV|USERdefine}:LINE	Sets or queries the line where TV triggering will be used.	5-282
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PAL|SDTV|USERdefine}:POLarity	Sets or queries the TV trigger input polarity.	5-282
:TRIGger[:ATRigger]:TV:LEVel	Sets or queries the TV trigger level.	5-282
:TRIGger[:ATRigger]:TV:LFORmat	Sets or queries the format for specifying the line where TV triggering will be used.	5-283
:TRIGger[:ATRigger]:TV:SOURce	Sets or queries the TV trigger source.	5-283
:TRIGger[:ATRigger]:TV:TYPE	Sets or queries the TV trigger input type.	5-283
:TRIGger[:ATRigger]:TV:USERdefine?	Queries all user-defined TV trigger settings.	5-283
:TRIGger[:ATRigger]:TV:USERdefine:DEFinition	Sets or queries the user-defined TV trigger definition setting.	5-283
:TRIGger[:ATRigger]:TV:USERdefine:HFRejection (HighFrequencyREJECTION)	Sets or queries the low-pass filter (HF rejection) setting for user-defined TV triggering.	5-283
:TRIGger[:ATRigger]:TV:USERdefine:HSYNc(Hsync Freq)	Sets or queries the horizontal sync frequency for user-defined TV triggering.	5-283
:TRIGger[:ATRigger]:TV:USERdefine:LINE	Sets or queries the user-defined TV trigger line number.	5-283
:TRIGger[:ATRigger]:TV:USERdefine:POLarity	Sets or queries the user-defined TV trigger input polarity.	5-284
:TRIGger[:ATRigger]:TV:USERdefine:SGUard	Sets or queries the user-defined TV trigger sync guard setting.	5-284
:TRIGger[:ATRigger]:WIDTh?	Queries all state width trigger settings.	5-284
:TRIGger[:ATRigger]:WIDTh:BITS?	Queries all logic input settings for state width triggering.	5-284
:TRIGger[:ATRigger]:WIDTh:BITS:HEXa	Sets the logic input state for state width triggering in hexadecimal notation.	5-284
:TRIGger[:ATRigger]:WIDTh:BITS:PATTern	Sets or queries the logic input state pattern for state width triggering.	5-284
:TRIGger[:ATRigger]:WIDTh:CHANnel<x>	Sets or queries the trigger condition of a channel for state width triggering.	5-284
:TRIGger[:ATRigger]:WIDTh:CLOCk?	Queries all clock signal settings for state width triggering.	5-284
:TRIGger[:ATRigger]:WIDTh:CLOCk:POLarity	Sets or queries the clock signal polarity for state width triggering.	5-284
:TRIGger[:ATRigger]:WIDTh:CLOCk:SOURce	Sets or queries the clock signal for state width triggering.	5-285
:TRIGger[:ATRigger]:WIDTh:CONDition	Sets or queries the trigger condition for state width triggering.	5-285
:TRIGger[:ATRigger]:WIDTh:LOGic	Sets or queries the combination condition for state width triggering.	5-285
:TRIGger[:ATRigger]:WIDTh:TIME<x>	Sets or queries the pulse width setting for state width triggering.	5-285
:TRIGger[:ATRigger]:WIDTh:TYPE	Sets or queries the time width mode for state width triggering.	5-285
:TRIGger:COMBination	Sets or queries the trigger combination.	5-285
:TRIGger:DELay?	Queries all delay settings.	5-285
:TRIGger:DELay:TIME	Sets or queries the delay (the time between the trigger point and the trigger position).	5-285
:TRIGger:HOLDoff?	Queries all hold-off settings.	5-285
:TRIGger:HOLDoff:TIME	Sets or queries the hold-off time.	5-285
:TRIGger:MODE	Sets or queries the trigger mode.	5-286
:TRIGger:POSition	Sets or queries the trigger position.	5-286
:TRIGger:SCOunt	Sets or queries the number of times the trigger condition must be met when the trigger mode is set to Single(N).	5-286

Command	Function	Page
:TRIGger:SOURce?	Queries all trigger source settings for enhanced triggering.	5-286
:TRIGger:SOURce:CHANnel<x>?	Queries all of the settings of the specified trigger source for enhanced triggering.	5-286
:TRIGger:SOURce:CHANnel<x>:COUPling	Sets or queries the specified trigger source trigger coupling for enhanced triggering.	5-286
:TRIGger:SOURce:CHANnel<x>:HFRejection (HighFrequencyREJECTION)	Sets or queries the low-pass filter setting of the specified trigger source for enhanced triggering.	5-286
:TRIGger:SOURce:CHANnel<x>:HYSTeresis	Sets or queries the noise rejection setting of the specified trigger source for enhanced triggering.	5-286
:TRIGger:SOURce:CHANnel<x>:LEVel	Sets or queries the trigger level of the specified trigger source for enhanced triggering.	5-286
:TRIGger:SOURce:CHANnel<x>:WIDTh	Sets or queries the window width of the specified trigger source for enhanced triggering.	5-287
:TRIGger:SOURce:CHANnel<x>:WINDow	Sets or queries the window on/off status of the specified trigger source for enhanced triggering.	5-287

WAVeform Group

:WAVeform?	Queries all information about waveform data.	5-288
:WAVeform:ALL?	Queries all settings related to the :WAVeform:ALL:SEND? query.	5-288
:WAVeform:ALL:SEND?	Queries the waveform data specified by the :WAVeform:ALL:TRACe command.	5-288
:WAVeform:ALL:TRACe	Sets or queries the waveform that is the target of the :WAVeform:ALL:SEND? query.	5-289
:WAVeform:BITS?	Queries the bit length of the waveform data specified by the :WAVeform:TRACe command.	5-289
:WAVeform:BYTeorder	Sets or queries the transmission byte order for waveform data in word format that is 2 bytes or longer in length.	5-289
:WAVeform:END	Sets or queries which point will be used as the last data value for the waveform specified by :WAVeform:TRACe.	5-289
:WAVeform:FORMat	Sets or queries the format of the waveform data to be sent.	5-289
:WAVeform:LENGth?	Queries the total number of data points of the waveform data specified by the :WAVeform:TRACe command.	5-289
:WAVeform:OFFSet?	Queries the offset value used to convert the waveform data specified by the :WAVeform:TRACe command to physical values.	5-289
:WAVeform:POSition?	Queries the vertical position that is used to convert waveform data to voltage when :WAVeform:FORMat is set to RBYTe.	5-289
:WAVeform:RANGe?	Queries the range used to convert the waveform data specified by the :WAVeform:TRACe command to physical values.	5-289
:WAVeform:RECord	Sets or queries the record number that WAVeform commands will be applied to.	5-289
:WAVeform:RECord? MINimum	Queries the minimum record number of the source channel.	5-290
:WAVeform:SEND?	Queries the waveform data specified by the :WAVeform:TRACe command.	5-290
:WAVeform:SIGN?	Queries whether or not signs are included in the binary data of the source waveform specified by :WAVeform:TRACe when the data is queried.	5-290
:WAVeform:SRATe? (Sample RATE)	Queries the sample rate of the record specified by the :WAVeform:RECord command.	5-290
:WAVeform:STARt	Sets or queries which point will be used as the first data value for the waveform specified by :WAVeform:TRACe.	5-290
:WAVeform:TRACe	Sets or queries the waveform that WAVeform commands will be applied to.	5-290
:WAVeform:TRIGger?	Queries the trigger position of the record specified by the :WAVeform:RECord command.	5-290
:WAVeform:TYPE?	Queries the acquisition mode of the waveform specified by the :WAVeform:TRACe command.	5-290

WPARameter Group

:WPARameter<x>?	Queries all of the settings for the waveform parameter measurement feature.	5-291
:WPARameter<x>:DISPlay	Sets or queries the on/off status of waveform parameter measurement display.	5-291
:WPARameter<x>:HISTogram?	Queries all histogram display settings for waveform parameter measurement.	5-291
:WPARameter<x>:HISTogram:MEASure?	Queries all automatic measurement settings of the histogram display for waveform parameter measurement.	5-291
:WPARameter<x>:HISTogram:MEASure:MODE	Sets or queries the automatic measurement mode of the histogram display for waveform parameter measurement.	5-291

Command	Function	Page
:WPARameter<x>:HISTogram:MEASure:PARameter?	Queries all automatic measurement settings of the histogram parameter for waveform parameter measurement.	5-291
:WPARameter<x>:HISTogram:MEASure:PARameter:ALL	Collectively turns on or off all histogram parameters for waveform parameter measurement.	5-291
:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>?	Queries all of the settings of a histogram parameter for waveform parameter measurement.	5-291
:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>:STATe	Sets or queries the on/off status of a histogram parameter for waveform parameter measurement.	5-292
:WPARameter<x>:HISTogram:MEASure:PARameter:<Parameter>:VALue?	Queries the measured value of a histogram parameter for waveform parameter measurement.	5-292
:WPARameter<x>:HISTogram:MEASure:PARameter:POSition<y>	Sets or queries the position of a histogram parameter for waveform parameter measurement.	5-292
:WPARameter<x>:ITEM	Sets or queries a waveform parameter for waveform parameter measurement.	5-292
:WPARameter<x>:MODE	Sets or queries the waveform parameter measurement mode.	5-292
:WPARameter<x>:TRENd?	Queries all trend display settings for waveform parameter measurement.	5-292
:WPARameter<x>:TRENd:ASCale	Executes auto scaling of the trend display for waveform parameter measurement.	5-292
:WPARameter<x>:TRENd:CURSor?	Queries all cursor measurement settings of the trend display for waveform parameter measurement.	5-293
:WPARameter<x>:TRENd:CURSor:C<y>?	Queries the measured value of a cursor in the trend display for waveform parameter measurement.	5-293
:WPARameter<x>:TRENd:CURSor:DC?	Queries the measured value between cursors in the trend display for waveform parameter measurement.	5-293
:WPARameter<x>:TRENd:CURSor:MODE	Sets or queries the automatic measurement mode of the trend display for waveform parameter measurement.	5-293
:WPARameter<x>:TRENd:CURSor:POSition<y>	Sets or queries the position of a cursor in the trend display for waveform parameter measurement.	5-293
:WPARameter<x>:TRENd:HRANge	Sets or queries the trend display source window for waveform parameter measurement.	5-293
:WPARameter<x>:TRENd:HSPan	Sets or queries the horizontal span of the trend display for waveform parameter measurement.	5-293
:WPARameter<x>:TRENd:VERTical	Sets or queries the vertical range of the trend display for waveform parameter measurement.	5-293
:WPARameter<x>:VTDisplay	Sets or queries the on/off status of VT waveform display.	5-294
:WPARameter<x>:WAIT?	Waits for the completion of automated measurement with a set timeout.	5-294

XY Group

Command	Function	Page
:XY<x>?	Queries all XY display settings.	5-295
:XY<x>:DISPlay	Sets or queries the on/off status of the XY display.	5-295
:XY<x>:MEASure?	Queries all automated measurement settings for the XY display.	5-295
:XY<x>:MEASure:CURSor?	Queries all cursor measurement settings for the XY display.	5-295
:XY<x>:MEASure:CURSor:DX?	Queries all settings for the voltage difference between the X-Y display's horizontal cursors.	5-295
:XY<x>:MEASure:CURSor:DX:STATe	Sets or queries the on/off status of the voltage difference between the X-Y display's horizontal cursors.	5-295
:XY<x>:MEASure:CURSor:DX:VALue?	Queries the voltage difference between the X-Y display's horizontal cursors.	5-295
:XY<x>:MEASure:CURSor:DY?	Queries all settings for the voltage difference between the X-Y display's vertical cursors.	5-295
:XY<x>:MEASure:CURSor:DY:STATe	Sets or queries the on/off status of the voltage difference between the X-Y display's vertical cursors.	5-295
:XY<x>:MEASure:CURSor:DY:VALue?	Queries the voltage difference between the X-Y display's vertical cursors.	5-295
:XY<x>:MEASure:CURSor:X<y>?	Queries all horizontal cursor settings for the XY display.	5-295
:XY<x>:MEASure:CURSor:X<y>:POSition	Sets or queries a horizontal cursor position in the XY display.	5-296
:XY<x>:MEASure:CURSor:X<y>:STATe	Sets or queries the on/off status of the voltage of an X-Y display horizontal cursor.	5-296
:XY<x>:MEASure:CURSor:X<y>:VALue?	Queries the voltage of a horizontal cursor in the XY display.	5-296
:XY<x>:MEASure:CURSor:Y<y>?	Queries all vertical cursor settings for the XY display.	5-296
:XY<x>:MEASure:CURSor:Y<y>:POSition	Sets or queries a vertical cursor position in the XY display.	5-296

Command	Function	Page
:XY<x>:MEASure:CURSor:Y<y>:STATe	Sets or queries the on/off status of the voltage of an X-Y display vertical cursor.	5-296
:XY<x>:MEASure:CURSor:Y<y>:VALue?	Queries the voltage of a vertical cursor in the XY display.	5-296
:XY<x>:MEASure:INTeg?	Queries all integration settings for the XY display.	5-296
:XY<x>:MEASure:INTeg:LOOP	Sets or queries the integration method for the XY display.	5-296
:XY<x>:MEASure:INTeg:POLarity	Sets or queries the integration polarity for the XY display.	5-297
:XY<x>:MEASure:INTeg:VALue?	Queries the integrated value for the XY display.	5-297
:XY<x>:MEASure:MODE	Sets or queries the automated measurement mode for the XY display.	5-297
:XY<x>:SPLit	Sets or queries whether or not the XY display are split to XY1 and XY2 windows.	5-297
:XY<x>:TRANge (Time Range)	Sets or queries the range of T-Y waveform to display in the XY display.	5-297
:XY<x>:VTDisplay	Sets or queries the on/off status of the VT waveform display for the XY display.	5-297
:XY<x>:XTRace	Sets or queries the channel that is assigned to the X-axis in the XY display.	5-297
:XY<x>:YTRace	Sets or queries the channel that is assigned to the Y-axis in the XY display.	5-297

ZOOM Group

:ZOOM<x>?	Queries all waveform zoom settings.	5-298
:ZOOM<x>:ALLocation?	Queries all zoom source waveform settings.	5-298
:ZOOM<x>:ALLocation:{CHANnel<y>|MATH<y>}	Sets or queries the zoom source waveform.	5-298
:ZOOM<x>:DISPlay	Sets or queries the on/off status of a zoom waveform display.	5-298
:ZOOM<x>:FORMat	Sets or queries the format of a zoom waveform display.	5-298
:ZOOM<x>:MAG	Sets or queries the magnification of a zoom waveform display.	5-298
:ZOOM<x>:MAGFine	Sets or queries the magnification (Fine) of a zoom waveform display.	5-298
:ZOOM<x>:MAIN	Sets or queries the display ratio of a zoom waveform display.	5-298
:ZOOM<x>:POSition	Sets or queries the position of a zoom box.	5-298
:ZOOM<x>:VERTical?	Queries all vertical zoom settings.	5-298
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>}?	Queries all of the settings of a trace for vertical zooming.	5-299
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>}:MAG	Sets or queries the vertical zoom factor.	5-299
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>}:POSition	Sets or queries the vertical zoom position.	5-299
:ZOOM<x>:VERTical:TRACe	Sets or queries the trace that is displayed in the vertical zoom window.	5-299

Common Command Group

*CAL? (CALibrate)	Executed calibration and queries the result.	5-300
*CLS (CLear Status)	Clears the standard event register, extended event register, and error queue.	5-300
*ESE (standard Event Status Enable register)	Sets or queries the standard event enable register.	5-300
*ESR? (standard Event Status Register)	Queries and clears the standard event register.	5-300
*IDN? (IDeNtify)	Queries the DLM2000 model.	5-300
*OPC (OPeration Complete)	Sets bit 0 (the OPC bit) of the standard event register to 1 upon the completion of the specified overlap command.	5-300
*OPC? (OPeration Complete)	If you send *OPC?, the DLM2000 returns ASCII code 1 when the specified overlap command is completed.	5-301
*OPT? (OPTion)	Queries the installed options.	5-301
*RST (ReSeT)	Resets the settings.	5-301
*SRE (Service Request Enable register)	Sets or queries the service request enable register value.	5-301
*STB? (STatus Byte)	Queries the status byte register value.	5-301
*TST?	Performs a self-test and queries the result. The self-test consists of tests of each kind of internal memory.	5-301
*WAI (WAIt)	Holds the subsequent command until the completion of the specified overlap operation.	5-302

5.2 ACQuire Group

:ACQuire?

Function Queries all waveform acquisition settings.

Syntax `:ACQuire?`

:ACQuire:AVERage?

Function Queries all of the settings related to averaging and the number of waveform acquisitions.

Syntax `:ACQuire:AVERage?`

:ACQuire:AVERage:COUNt

Function Sets or queries the attenuation constant for exponential averaging or the average count for linear averaging for Averaging mode.

Syntax `:ACQuire:AVERage:COUNt {<NRf>}`

<NRf> = 2 to 1024 in 2^n steps)

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

:ACQuire:COUNt

Function Sets or queries the number of waveform acquisitions 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:INTerleave

Function Sets or queries the on/off status of Interleave mode.

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

Example
```
:ACQUIRE:INTERLEAVE ON
:ACQUIRE:INTERLEAVE?
-> :ACQUIRE:INTERLEAVE 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 on/off status of High Resolution mode.

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

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

Description The maximum record length is reduced to half when High Resolution mode is turned on.

:ACQuire:RLENgth

Function Sets or queries the record length.

Syntax `:ACQuire:RLENgth {<NRf>}`

<NRf> = See the DLM2000 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 of the settings for the analysis feature.

Syntax `:ANALysis?`

:ANALysis:AHIStogram<x>?

Function Queries all of the settings for the waveform histogram feature.

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

:ANALysis:AHIStogram<x>:DISPlay

Function Sets or queries the on/off status of a waveform histogram display.

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 a waveform histogram.

Syntax
```
:ANALysis:AHIStogram<x>:HORizontal {<NRf>,<NRf>}
:ANALysis:AHIStogram<x>:HORizontal?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = −4 to 4 (divisions)

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

:ANALysis:AHIStogram<x>:MEASure?

Function Queries all of the settings for waveform histogram measurement (including the mode on/off status).

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 measurement mode of a waveform histogram.

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

Example
```
:ANALYSIS:AHISTOGRAM1:MEASURE:MODE OFF
:ANALYSIS:AHISTOGRAM1:MEASURE:MODE?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:MODE OFF
```

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

Function Queries all of the measurement settings for a waveform histogram when the histogram parameter measurement mode is set to Param.

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

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

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

Syntax `:ANALysis:AHIStogram<x>:MEAS ure:PARameter:ALL {<Boolean>}`
<x> = 1 or 2 (1 only for 2-channel models)

Example `:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMETER:ALL ON`

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

Function Queries all of the settings for the specified measurement item of a waveform histogram.

Syntax `:ANALysis:AHIStogram<x>: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`}

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

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

Syntax
```
:ANALysis:AHIStogram<x>:MEASure:PARam
eter:<Parameter>:STATe {<Boolean>}
:ANALysis:AHIStogram<x>:MEASure:PARam
eter:<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 Below are examples for the maximum value.
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMET
ER:MAXIMUM:STATE ON
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMET
ER:MAXIMUM:STATE?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARA
METER:MAXIMUM:STATE 1
```

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

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

Syntax
```
:ANALysis:AHIStogram<x>:MEASure:PARam
eter:<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 Below are examples for the maximum value.
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMET
ER:MAXIMUM:VALUE?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARA
METER:MAXIMUM:VALUE 1.000E+00
```

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

Function Sets or queries the horizontal position of Cursor1 or Cursor 2 for a waveform histogram.

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

Example
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMET
ER:HPOSITION1 1
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMET
ER:HPOSITION1?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARA
METER:HPOSITION 1.000E+00
```

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

Function Sets or queries the vertical position of Cursor1 or Cursor 2 for a waveform histogram.

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

Example
```
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMET
ER:VPOSITION1 1
:ANALYSIS:AHISTOGRAM1:MEASURE:PARAMET
ER:VPOSITION1?
-> :ANALYSIS:AHISTOGRAM1:MEASURE:PARA
METER:VPOSITION 1.000E+00
```

:ANALysis:AHIStogram<x>:MODE

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

Syntax
```
:ANALysis:AHIStogram<x>:MODE {HORizon
tal|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 HORIZON
TAL
```

:ANALysis:AHIStogram<x>:RANGe

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

Syntax
```
:ANALysis:AHIStogram<x>:RANGe {MAIN|Z
1|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 a 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 for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 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 a 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 to 4 (divisions)

Example
```
:ANALYSIS:AHISTOGRAM1:VERTICAL 0,1
:ANALYSIS:AHISTOGRAM1:VERTICAL?
-> :ANALYSIS:AHISTOGRAM1:VERTICAL 1.0
00E+00,0.000E+00
```

:ANALysis:PANalyze<x>?

Function Queries all power supply analysis settings.

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

:ANALysis:PANalyze<x>:I2T?

Function Queries all Joule integral settings.

Syntax
```
:ANALysis:PANalyze<x>:I2T?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:I2T:MATH

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

Syntax
```
:ANALysis:PANalyze<x>:I2T:MATH {I2T|O
FF}
:ANALysis:PANalyze<x>:I2T:MATH?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:I2T:MATH I2T
:ANALYSIS:PANALYZE1:I2T:MATH?
-> :ANALYSIS:PANALYZE1:I2T:MATH I2T
```

:ANALysis:PANalyze<x>:I2T:MEASure?

Function Queries all Joule integral automatic measurement settings.

Syntax
```
:ANALysis:PANalyze<x>:I2T:MEASure?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:I2T:MEASure:I2T?

Function Queries Joule integral settings.

Syntax
```
:ANALysis:PANalyze<x>:I2T:MEASure:I
2T?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:I2T:MEASure:I2T:COUNt?

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

Syntax
```
:ANALysis:PANalyze<x>:I2T:MEASure:I2T
:COUNt?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:I2T:MEASURE:I2T:C
OUNT?
-> :ANALYSIS:PANALYZE1:I2T:MEASURE:I2
T:COUNT 100
```

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

Function Queries a statistical value of a Joule integral.

Syntax
```
:ANALysis:PANalyze<x>:I2T:MEASure:I2T
:{MAXimum|MEAN|MINimum|
SDEViation}?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:I2T:MEASURE:I2T:M
AXIMUM?
-> :ANALYSIS:PANALYZE1:I2T:MEASURE:I2
T:MAXIMUM 10.0000E+03
```

Description If the statistical value is immeasurable, the DLM2000 returns “NAN” (not a number).

:ANALysis:PANalyze<x>:I2T:MEASure:I2T:STATe

Function Sets or queries whether or not a Joule integral will be measured.

Syntax
```
:ANALysis:PANalyze<x>:I2T:MEASure:I2T
:STATe {<Boolean>}
:ANALysis:PANalyze<x>:I2T:MEASure:I2T
:STATe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:I2T:MEASURE:I2T:S
TATE ON
:ANALYSIS:PANALYZE1:I2T:MEASURE:I2T:S
TATE?
-> :ANALYSIS:PANALYZE1:I2T:MEASURE:I2
T:STATE 1
```

:ANALysis:PANalyze<x>:I2T:MEASure:I2T:VALue?

Function Queries an automatically measured value of a Joule integral.

Syntax
```
:ANALysis:PANalyze<x>:I2T:MEASure:I2T
:VALue? [{<NRf>}]
```
<x> = 1 or 2

<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE1:I2T:MEASURE:I2T:V
ALUE?
-> :ANALYSIS:PANALYZE1:I2T:MEASURE:I2
T:VALUE 10.0000E+03
```

Description
- If the value is immeasurable, the DLM2000 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 DLM2000 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<x>:I2T:RANGe

Function Sets or queries the measurement source window.

Syntax
```
:ANALysis:PANalyze<x>:I2T:RANGe {MAIN
|Z1|Z2}
:ANALysis:PANalyze<x>:I2T:RANGe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:I2T:RANGE MAIN
:ANALYSIS:PANALYZE1:I2T:RANGE?
-> :ANALYSIS:PANALYZE1:I2T:RANGE MAIN
```

:ANALysis:PANalyze<x>:I2T:SCALe?

Function Queries all scaling settings.

Syntax
```
:ANALysis:PANalyze<x>:I2T:SCALe?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:I2T:SCALe:CENTer

Function Sets or queries the center value for manual scaling.

Syntax
```
:ANALysis:PANalyze<x>:I2T:SCALe:CENTer {<NRf>}
:ANALysis:PANalyze<x>:I2T:SCALe:CENTer?
```
<x> = 1 or 2
<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE1:I2T:SCALE:CENTER 1
:ANALYSIS:PANALYZE1:I2T:SCALE:CENTER?
-> :ANALYSIS:PANALYZE1:I2T:SCALE:CENTER 1.00000E+00
```

:ANALysis:PANalyze<x>:I2T:SCALe:MODE

Function Sets or queries the scaling mode.

Syntax
```
:ANALysis:PANalyze<x>:I2T:SCALe:MODE {AUTO|MANual}
:ANALysis:PANalyze<x>:I2T:SCALe:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:I2T:SCALE:MODE AUTO
:ANALYSIS:PANALYZE1:I2T:SCALE:MODE?
-> :ANALYSIS:PANALYZE1:I2T:SCALE:MODE AUTO
```

:ANALysis:PANalyze<x>:I2T:SCALe:SENSitivity

Function Sets or queries the span from the center value for manual scaling.

Syntax
```
:ANALysis:PANalyze<x>:I2T:SCALe:SENSitivity {<NRf>}
:ANALysis:PANalyze<x>:I2T:SCALe:SENSitivity?
```
<x> = 1 or 2
<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE1:I2T:SCALE:SENSITIVITY 10
:ANALYSIS:PANALYZE1:I2T:SCALE:SENSITIVITY?
-> :ANALYSIS:PANALYZE1:I2T:SCALE:SENSITIVITY 10.0000E+00
```

:ANALysis:PANalyze<x>:I2T:TRANge (Time Range)

Function Sets or queries the measurement time period.

Syntax
```
:ANALysis:PANalyze<x>:I2T:TRANge {<NRf>,<NRf>}
:ANALysis:PANalyze<x>:I2T:TRANge?
```
<x> = 1 or 2
<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:ANALYSIS:PANALYZE1:I2T:TRANGE -4,4
:ANALYSIS:PANALYZE1:I2T:TRANGE?
-> :ANALYSIS:PANALYZE1:I2T:TRANGE -4.00E+00,4.00E+00
```

:ANALysis:PANalyze<x>:HARMonics?

Function Queries all harmonic analysis settings.

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

:ANALysis:PANalyze<x>:HARMonics:CCLass?

Function Queries all class C harmonic analysis settings.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:CCLass?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:HARMonics:CCLass:GETLambda

Function Queries the current power factor for class C harmonic analysis.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:CCLass:GETLambda
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:CCLASS:GETLAMBDA
```

:ANALysis:PANalyze<x>:HARMonics:CCLass:LAMBda

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

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:CCLass:LAMBda {<NRf>}
:ANALysis:PANalyze<x>:HARMonics:CCLass:LAMBda?
```
<x> = 1 or 2
<NRf> = 0.001 to 1

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:CCLASS:LAMBDA 0.10
:ANALYSIS:PANALYZE1:HARMONICS:CCLASS:LAMBDA?
-> :ANALYSIS:PANALYZE1:HARMONICS:CCLASS:LAMBDA 100.0E-03
```

:ANALysis:PANalyze<x>:HARMonics:CCLass:MAXCurrent

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

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:CCLass:MAXCurrent {<NRf>|<Current>}
:ANALysis:PANalyze<x>:HARMonics:CCLass:MAXCurrent?
```
<x> = 1 or 2

<NRf> or <Current> = 0.001 to 100 (A)

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:CCLASS:MAXCURRENT 50A
:ANALYSIS:PANALYZE1:HARMONICS:CCLASS:MAXCURRENT?
-> :ANALYSIS:PANALYZE1:HARMONICS:CCLASS:MAXCURRENT 50.000E+00
```

:ANALysis:PANalyze<x>:HARMonics:CCLass:OPOWer

Function Sets or queries whether or not the active power for class C harmonic analysis exceeds 25 W.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:CCLass:OPOWer {FALSe|TRUE}
:ANALysis:PANalyze<x>:HARMonics:CCLass:OPOWer?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:CCLASS:OPOWER FALSE
:ANALYSIS:PANALYZE1:HARMONICS:CCLASS:OPOWER?
-> :ANALYSIS:PANALYZE1:HARMONICS:CCLASS:OPOWER FALSE
```

:ANALysis:PANalyze<x>:HARMonics:CLASs

Function Sets or queries the class of the device under harmonic analysis.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:CLASs {A|B|C|D}
:ANALysis:PANalyze<x>:HARMonics:CLASs?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:CLASS A
:ANALYSIS:PANALYZE1:HARMONICS:CLASS?
-> :ANALYSIS:PANALYZE1:HARMONICS:CLASS A
```

:ANALysis:PANalyze<x>:HARMonics:DCLass?

Function Queries all class D harmonic analysis settings.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:DCLass?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:HARMonics:DCLass:POWer

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

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:DCLass:POWer {<NRf>}
:ANALysis:PANalyze<x>:HARMonics:DCLass:POWer?
```
<x> = 1 or 2

<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:DCLASS:POWER 1V
:ANALYSIS:PANALYZE1:HARMONICS:DCLASS:POWER?
-> :ANALYSIS:PANALYZE1:HARMONICS:DCLASS:POWER 1.000E+00
```

:ANALysis:PANalyze<x>:HARMonics:DMODe

Function Sets or queries the harmonic analysis display mode.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:DMODe {LINear|LOG}
:ANALysis:PANalyze<x>:HARMonics:DMODe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:DMODE LINEAR
:ANALYSIS:PANALYZE1:HARMONICS:DMODE?
-> :ANALYSIS:PANALYZE1:HARMONICS:DMODE LINEAR
```

:ANALysis:PANalyze<x>:HARMonics:DETail?

Function Queries all harmonic analysis result list settings.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:DETail?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:HARMonics:DETail:DISPlay

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

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:DETail:DISPlay {FULL|LOWer|UPPer}
:ANALysis:PANalyze<x>:HARMonics:DETail:DISPlay?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:DETAIL:DISPLAY FULL
:ANALYSIS:PANALYZE1:HARMONICS:DETAIL:DISPLAY?
-> :ANALYSIS:PANALYZE1:HARMONICS:DETAIL:DISPLAY FULL
```

:ANALysis:PANalyze<x>:HARMonics:DETail:LIST:ITEM?

Function Queries the items displayed in the harmonic analysis result list.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:DETail:LIST:ITEM?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:DETAIL:LIST:ITEM?
-> :ANALYSIS:PANALYZE1:HARMONICS:DETAIL:LIST:ITEM "Order.,Measure(A),Limit(A),Measure(%),Limit(%),Info,"
```

:ANALysis:PANalyze<x>:HARMonics:DETail:LIST:VALue?

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

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:DETail:LIST:VALue? {<NRf>}
```
<x> = 1 or 2

<NRf> = 1 to 40 (harmonic order)

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:DETAIL:LIST:VALUE? 2
-> :ANALYSIS:PANALYZE1:HARMONICS:DETAIL:LIST:VALUE " 2, 0.031, 0.020, 3.149, 2.000,NG,"
```

:ANALysis:PANalyze<x>:HARMonics:GROuping

Function Sets or queries the harmonic analysis grouping.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:GROuping {OFF|TYPE1|TYPE2}
:ANALysis:PANalyze<x>:HARMonics:GROuping?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:GROUPING OFF
:ANALYSIS:PANALYZE1:HARMONICS:GROUPING?
-> :ANALYSIS:PANALYZE1:HARMONICS:GROUPING OFF
```

:ANALysis:PANalyze<x>:HARMonics:SPOint

Function Sets or queries the harmonic analysis computation start point.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:SPOint {<NRf>}
:ANALysis:PANalyze<x>:HARMonics:SPOint?
```
<x> = 1 or 2

<NRf> = −5 to 5 (divisions)

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:SPOINT 1
:ANALYSIS:PANALYZE1:HARMONICS:SPOINT?
-> :ANALYSIS:PANALYZE1:HARMONICS:SPOINT 1.000E+00
```

:ANALysis:PANalyze<x>:HARMonics:SVOLtage

Function Sets or queries the power supply voltage for harmonic analysis.

Syntax
```
:ANALysis:PANalyze<x>:HARMonics:SVOLtage {<NRf>|<Current>}
:ANALysis:PANalyze<x>:HARMonics:SVOLtage?
```
<x> = 1 or 2

<NRf>, <Voltage> = 90 to 440 V

Example
```
:ANALYSIS:PANALYZE1:HARMONICS:SVOLTAGE 230
:ANALYSIS:PANALYZE1:HARMONICS:SVOLTAGE?
-> :ANALYSIS:PANALYZE1:HARMONICS:SVOLTAGE 220.00000E+00
```

:ANALysis:PANalyze<x>:SETup?

Function Queries all power supply analysis input settings.

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

:ANALysis:PANalyze<x>:SETup:ADESkew

Function Executes automatic deskewing for power supply analysis.

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

Example
```
:ANALYSIS:PANALYZE1:SETUP:ADESKEW
```

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

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

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

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

Function Sets or queries the deskew setting of the current input channel for power supply analysis.

Syntax
```
:ANALysis:PANalyze<x>:SETup:I:DESK
ew {<Time>}
:ANALysis:PANalyze<x>:SETup:I:DESKew?
```
<x> = 1 or 2
<Time> = −100.0 ns to 100.0 ns (in 10-ps steps)

Example
```
:ANALYSIS:PANALYZE1:SETUP:I:DESK
EW 1NS
:ANALYSIS:PANALYZE1:SETUP:I:DESKEW?
-> :ANALYSIS:PANALYZE1:SETUP:I:DESK
EW 1.000E-09
```

:ANALysis:PANalyze<x>:SETup:I:INPut

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

Syntax
```
:ANALysis:PANalyze<x>:SETup:I:INP
ut {2|4}
:ANALysis:PANalyze<x>:SETup:I:INPut?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SETUP:I:INPUT 2
:ANALYSIS:PANALYZE1:SETUP:I:INPUT?
-> :ANALYSIS:PANALYZE1:SETUP:I:
INPUT 2
```

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

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

Syntax
```
:ANALysis:PANalyze<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|
C20|C50|C100|C200|C500|C1000|C2000}}
:ANALysis:PANalyze<x>:SETup:I:PROBe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SETUP:I:PROBE C0_
001
:ANALYSIS:PANALYZE1:SETUP:I:PROBE?
-> :ANALYSIS:PANALYZE1:SETUP:I:
PROBE C0_001
```

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

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

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

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

Function Sets or queries the deskew setting of the voltage input channel for power supply analysis.

Syntax
```
:ANALysis:PANalyze<x>:SETup:U:DESK
ew {<Time>}
:ANALysis:PANalyze<x>:SETup:U:DESKew?
```
<x> = 1 or 2
<Time> = −100.0 ns to 100.0 ns (in 10-ps steps)

Example
```
:ANALYSIS:PANALYZE1:SETUP:U:DESK
EW 1NS
:ANALYSIS:PANALYZE1:SETUP:U:DESKEW?
-> :ANALYSIS:PANALYZE1:SETUP:U:DESK
EW 1.000E-09
```

:ANALysis:PANalyze<x>:SETup:U:INPut

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

Syntax
```
:ANALysis:PANalyze<x>:SETup:U:INP
ut {1|3}
:ANALysis:PANalyze<x>:SETup:U:INPut?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SETUP:U:INPUT 1
:ANALYSIS:PANALYZE1:SETUP:U:INPUT?
-> :ANALYSIS:PANALYZE1:SETUP:U:
INPUT 1
```

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

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

Syntax
```
:ANALysis:PANalyze<x>:SETup:U:PRO
Be {<NRf>}
:ANALysis:PANalyze<x>:SETup:U:PROBe?
```
<x> = 1 or 2
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE1:SETUP:U:PROBE 1
:ANALYSIS:PANALYZE1:SETUP:U:PROBE?
-> :ANALYSIS:PANALYZE1:SETUP:U:
PROBE 1.000
```

:ANALysis:PANalyze<x>:SETup:RTRace

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

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

Example
```
:ANALYSIS:PANALYZE1:SETUP:RTRACE I
:ANALYSIS:PANALYZE1:SETUP:RTRACE?
-> :ANALYSIS:PANALYZE1:SETUP:RTRACE I
```

:ANALysis:PANalyze<x>:SOA?

Function Queries all XY display (safe operating area) settings.

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

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

Function Queries all XY display (safe operating area) cursor measurement settings.

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

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

Function Queries all XY display (safe operating area) horizontal cursor settings.

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

:ANALysis:PANalyze<x>:SOA:CURSor:X<y>:POSition

Function Sets or queries a horizontal cursor position in the XY display (safe operating area).

Syntax
```
:ANALysis:PANalyze<x>:SOA:CURSor:X<y>:POSition {<NRf>}
:ANALysis:PANalyze<x>:SOA:CURSor:X<y>:POSition?
```
<x> = 1 or 2
<y> = 1 or 2
<NRf> = −4 to 4 (divisions)

Example
```
:ANALYSIS:PANALYZE1:SOA:CURSOR:X1:POSITION 1
:ANALYSIS:PANALYZE1:SOA:CURSOR:X1:POSITION?
-> :ANALYSIS:PANALYZE1:SOA:CURSOR:X:POSITION 1.000E+00
```

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

Function Sets or queries the voltage value of a horizontal cursor in the XY display (safe operating area).

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

Example
```
:ANALYSIS:PANALYZE1:SOA:CURSOR:X1:VALUE?
-> :ANALYSIS:PANALYZE1:SOA:CURSOR:X1:VALUE 1.000E+00
```

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

Function Queries all XY display (safe operating area) vertical cursor settings.

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

:ANALysis:PANalyze<x>:SOA:CURSor:Y<y>:POSition

Function Sets or queries a vertical cursor position in the XY display (safe operating area).

Syntax
```
:ANALysis:PANalyze<x>:SOA:CURSor:Y<y>:POSition {<NRf>}
:ANALysis:PANalyze<x>:SOA:CURSor:Y<y>:POSition?
```
<x> = 1 or 2
<y> = 1 or 2
<NRf> = −4 to 4 (divisions)

Example
```
:ANALYSIS:PANALYZE1:SOA:CURSOR:Y1:POSITION 1
:ANALYSIS:PANALYZE1:SOA:CURSOR:Y1:POSITION?
-> :ANALYSIS:PANALYZE1:SOA:CURSOR:Y:POSITION 1.000E+00
```

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

Function Sets or queries the voltage value of a vertical cursor in the XY display (safe operating area).

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

Example
```
:ANALYSIS:PANALYZE1:SOA:CURSOR:Y1:VALUE?
-> :ANALYSIS:PANALYZE1:SOA:CURSOR:Y1:VALUE 1.000E+00
```

:ANALysis:PANalyze<x>:SOA:MODE

Function Sets or queries the automated measurement mode of the XY display (safe operating area).

Syntax
```
:ANALysis:PANalyze<x>:SOA:MODE {CURSo
r|OFF}
:ANALysis:PANalyze<x>:SOA:MODE?
```
<x> = 1 or 2

Example :ANALYSIS:PANALYZE1:SOA:MODE CURSOR
```
:ANALYSIS:PANALYZE1:SOA:MODE?
-> :ANALYSIS:PANALYZE1:SOA:MODE CURSO
R
```

:ANALysis:PANalyze<x>:SOA:TRANge (Time Range)

Function Sets or queries the range of the T-Y waveform to display in the XY display (safe operating area).

Syntax
```
:ANALysis:PANalyze<x>:SOA:TRAN
ge {<NRf>,<NRf>}
:ANALysis:PANalyze<x>:SOA:TRANge?
```
<x> = 1 or 2

<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:ANALYSIS:PANALYZE1:SOA:TRANGE -4,4
:ANALYSIS:PANALYZE1:SOA:TRANGE?
-> :ANALYSIS:PANALYZE1:SOA:TRANGE -4.
00,4.00
```

:ANALysis:PANalyze<x>:SOA:VTDisplay

Function Sets or queries whether or not to display the VT waveform in the XY display (safe operating area).

Syntax
```
:ANALysis:PANalyze<x>:SOA:VTDispl
ay {<Boolean>}
:ANALysis:PANalyze<x>:SOA:VTDisplay?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SOA:VTDISPLAY ON
:ANALYSIS:PANALYZE1:SOA:VTDISPLAY?
-> :ANALYSIS:PANALYZE1:SOA:
VTDISPLAY 1
```

:ANALysis:PANalyze<x>:SWLoss?

Function Queries all switching loss settings.

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

Description Use the commands listed below to set the reference levels (distal, mesial, proximal, etc.) used to determine the total loss.
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoxim
al?
:MEASure:{CHANnel<x>|MATH<x>}:DPRoxim
al:MODE
:MEASure:{CHANnel<x>|MATH<x>}:DPRoxim
al:PERCent
:MEASure:{CHANnel<x>|MATH<x>}:DPRoxim
al:UNIT
:MEASure:{CHANnel<x>|MATH<x>}:METHod
```

:ANALysis:PANalyze<x>:SWLoss:CYCLe

Function Sets or queries the on/off status of cycle mode.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:CYC
Le {<Boolean>}
:ANALysis:PANalyze<x>:SWLoss:CYCLe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:CYCLE ON
:ANALYSIS:PANALYZE1:SWLOSS:CYCLE?
-> :ANALYSIS:PANALYZE1:SWLOSS:CYCLE 1
```

:ANALysis:PANalyze<x>:SWLoss:DPROximal?

Function Queries all distal, mesial, and proximal settings.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:DPROxim
al?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL?
```

:ANALysis:PANalyze<x>:SWLoss:DPROximal:MODE

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

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:DPROxima
l:MODE {PERCent|UNIT}
:ANALysis:PANalyze<x>:SWLoss:DPROxima
l:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:
MODE PERCENT
:ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:
MODE?
-> :ANALYSIS:PANALYZE1:SWLOSS:DPROXIM
AL:MODE PERCENT
```

:ANALysis:PANalyze<x>:SWLoss:DPROximal:PERCent

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

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

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:PERCENT 10,50,90
:ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:PERCENT?
-> :ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:PERCENT 10,50,90
```

:ANALysis:PANalyze<x>:SWLoss:DPROximal:UNIT

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

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:DPROximal:UNIT {<NRf>,<NRf>,<NRf>}
:ANALysis:PANalyze<x>:SWLoss:DPROximal:UNIT?
```
<NRf>, <NRf>, <NRf> = See the DLM2000 Features Guide for this information.
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:UNIT -1,0,1
:ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:UNIT?
-> :ANALYSIS:PANALYZE1:SWLOSS:DPROXIMAL:UNIT -1.0000000E+00,
0.0000000E+00,1.0000000E+00
```

:ANALysis:PANalyze<x>:SWLoss:DTYPe

Function Sets or queries the device type for total loss computation.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:DTYPe {IGBT|MOSFET|OFF}
:ANALysis:PANalyze<x>:SWLoss:DTYPe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:DTYPE IGBT
:ANALYSIS:PANALYZE1:SWLOSS:DTYPE?
-> :ANALYSIS:PANALYZE1:SWLOSS:DTYPE
```

:ANALysis:PANalyze<x>:SWLoss:ILEVel

Function Sets or queries the current level used to determine the zero loss period for total loss computation.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:ILEVel {<NRf>|<Current>}
:ANALysis:PANalyze<x>:SWLoss:ILEVel?
```
<x> = 1 or 2
<NRf>, <Current> = See the DLM2000 Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:ILEVEL 1
:ANALYSIS:PANALYZE1:SWLOSS:ILEVEL?
-> :ANALYSIS:PANALYZE1:SWLOSS:ILEVEL 1.0000000E+00
```

:ANALysis:PANalyze<x>:SWLoss:MATH

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

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:MATH {OFF|POWer}
:ANALysis:PANalyze<x>:SWLoss:MATH?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:MATH OFF
:ANALYSIS:PANALYZE1:SWLOSS:MATH?
-> :ANALYSIS:PANALYZE1:SWLOSS:MATH OFF
```

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

Function Queries all of the settings for the automated measurement of power supply analysis parameters.

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

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

Function Queries the setting of a power supply analysis parameter.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>?
```
<x> = 1 or 2
<Parameter> = {P|PABS|PN|PP|WH|WHABs|WHN|WHP|Z}

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

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

Syntax `:ANALysis:PANalyze<x>:SWLoss:MEASure:P:COUNt?`
<x> = 1 or 2
<Parameter> = {`P|PABS|PN|PP|WH|WHABs|WHN|WHP|Z`}

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:MEASURE:P:
COUNT?
-> :ANALYSIS:PANALYZE1:SWLOSS:MEASURE
:P:COUNT 100
```

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

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

Syntax `:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?`
<x> = 1 or 2
<Parameter> = `{P|PABS|PN|PP|WH|WHABs|WHN|WHP|Z}`

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:MEASURE:P:
MAXIMUM?
-> :ANALYSIS:PANALYZE1:SWLOSS:MEASURE
:P:MAXIMUM 1.000E+00
```

Description If the statistical value is immeasurable, the DLM2000 returns “NAN” (not a number).

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

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

Syntax `:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:STATe {<Boolean>}`
`:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:STATe?`
<x> = 1 or 2
<Parameter> = `{P|PABS|PN|PP|WH|WHABs|WHN|WHP|Z}`

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:MEASURE:P:
STATE ON
:ANALYSIS:PANALYZE1:SWLOSS:MEASURE:P:
STATE?
-> :ANALYSIS:PANALYZE1:SWLOSS:MEASURE
:P:STATE 1
```

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

Function Queries an automatically measured value of a power supply analysis parameter.

Syntax `:ANALysis:PANalyze<x>:SWLoss:MEASure:<Parameter>:VALue? [{<NRf>}]`
<x> = 1 or 2
<NRf> = See the DLM2000 Features Guide for this information.
<Parameter> = `{P|PABS|PN|PP|WH|WHABs|WHN|WHP|Z}`

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:MEASURE:P:
VALUE?
-> :ANALYSIS:PANALYZE1:SWLOSS:MEASURE
:P:VALUE 10.0000E+03
```

Description
- If the value is immeasurable, the DLM2000 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 DLM2000 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<x>:SWLoss:METhod

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

Syntax `:ANALysis:PANalyze<x>:SWLoss:METhod {AUTO|MAXimum|HISTogram}`
`:ANALysis:PANalyze<x>:SWLoss:METhod?`
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:METHOD
AUTO
:ANALYSIS:PANALYZE1:SWLOSS:METHOD?
-> :ANALYSIS:PANALYZE1:SWLOSS:METHOD
AUTO
```

:ANALysis:PANalyze<x>:SWLoss:RANGe

Function Sets or queries the measurement source window.

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

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:RANGE MAIN
:ANALYSIS:PANALYZE1:SWLOSS:RANGE?
-> :ANALYSIS:PANALYZE1:SWLOSS:RANGE MAIN
```

:ANALysis:PANalyze<x>:SWLoss:RDS

Function Sets or queries the on-resistance value for total loss computation.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:RDS {<NRf>}
:ANALysis:PANalyze<x>:SWLoss:RDS?
```
<x> = 1 or 2

<NRf> = 0 to 100 (in 1 mΩ steps)

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:RDS 1
:ANALYSIS:PANALYZE1:SWLOSS:RDS?
-> :ANALYSIS:PANALYZE1:SWLOSS:RDS 1.000E+00
```

:ANALysis:PANalyze<x>:SWLoss:SCALe?

Function Queries all scaling settings.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:SCALe?
```
<x> = 1 or 2

:ANALysis:PANalyze<x>:SWLoss:SCALe:CENTer

Function Sets or queries the center value for manual scaling.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:SCALe:CENTer {<NRf>}
:ANALysis:PANalyze<x>:SWLoss:SCALe:CENTer?
```
<x> = 1 or 2

<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:SCALE:CENTER 1
:ANALYSIS:PANALYZE1:SWLOSS:SCALE:CENTER?
-> :ANALYSIS:PANALYZE1:SWLOSS:SCALE:CENTER 1.00000E+00
```

:ANALysis:PANalyze<x>:SWLoss:SCALe:MODE

Function Sets or queries the scaling mode.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:SCALe:MODE {AUTO|MANual}
:ANALysis:PANalyze<x>:SWLoss:SCALe:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:SCALE:MODE AUTO
:ANALYSIS:PANALYZE1:SWLOSS:SCALE:MODE?
-> :ANALYSIS:PANALYZE1:SWLOSS:SCALE:MODE AUTO
```

:ANALysis:PANalyze<x>:SWLoss:SCALe:SENSitivity

Function Sets or queries the span from the center value for manual scaling.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:SCALe:SENSitivity {<NRf>}
:ANALysis:PANalyze<x>:SWLoss:SCALe:SENSitivity?
```
<x> = 1 or 2

<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:SCALE:SENSITIVITY 10
:ANALYSIS:PANALYZE1:SWLOSS:SCALE:SENSITIVITY?
-> :ANALYSIS:PANALYZE1:SWLOSS:SCALE:SENSITIVITY 10.0000E+00
```

:ANALysis:PANalyze<x>:SWLoss:TRANge (Time Range)

Function Sets or queries the measurement time period.

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

<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:TRANGE -4,4
:ANALYSIS:PANALYZE1:SWLOSS:TRANGE?
-> :ANALYSIS:PANALYZE1:SWLOSS:TRANGE -4.00E+00,4.00E+00
```

:ANALysis:PANalyze<x>:SWLoss:ULEVel

Function Sets or queries the voltage level used to determine the loss calculation period for total loss computation.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:ULEV
el {<NRf>|<Voltage>}
:ANALysis:PANalyze<x>:SWLoss:ULEVel?
```
<x> = 1 or 2

<NRf>, <Voltage> = See the DLM2000 Features Guide for this information.

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:ULEVEL 1V
:ANALYSIS:PANALYZE1:SWLOSS:ULEVEL?
-> :ANALYSIS:PANALYZE1:SWLOSS:
ULEVEL 1.0000000E+00
```

:ANALysis:PANalyze<x>:SWLoss:UNIT

Function Sets or queries the power unit.

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

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:UNIT WH
:ANALYSIS:PANALYZE1:SWLOSS:UNIT?
-> :ANALYSIS:PANALYZE1:SWLOSS:UNIT WH
```

:ANALysis:PANalyze<x>:SWLoss:VCE

Function Sets or queries the collector-emitter saturation voltage value for total loss computation.

Syntax
```
:ANALysis:PANalyze<x>:SWLoss:V
CE {<NRf>|<Voltage>}
:ANALysis:PANalyze<x>:SWLoss:VCE?
```
<x> = 1 or 2

<NRf>, <Voltage> = 0 to 50 V (in 100-mV steps)

Example
```
:ANALYSIS:PANALYZE1:SWLOSS:VCE 1V
:ANALYSIS:PANALYZE1:SWLOSS:VCE?
-> :ANALYSIS:PANALYZE1:SWLOSS:VCE 1.0
000000E+00
```

:ANALysis:PANalyze<x>:TYPE

Function Sets or queries the type of power supply analysis.

Syntax
```
:ANALysis:PANalyze<x>:TYPE {HARMonics
|I2T|OFF|SOA|SWLoss}
:ANALysis:PANalyze<x>:TYPE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:PANALYZE1:TYPE HARMONICS
:ANALYSIS:PANALYZE1:TYPE?
-> :ANALYSIS:PANALYZE1:TYPE HARMONICS
```

:ANALysis:PMEAsure<x>?

Function Queries all power measurement settings.

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

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

:ANALysis:PMEAsure<x>:IDPRoximal?

Function Queries all distal, mesial, and proximal settings.

Syntax
```
:ANALysis:PMEAsure<x>:IDPRoximal?
```
<x> = 1 to 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 to 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:PERC
ent {<NRf>,<NRf>,<NRf>}
:ANALysis:PMEAsure<x>:IDPRoximal:PERC
ent?
```
<x> = 1 to 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:PER
CENT 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:UNIT {<NRf>,<NRf>,<NRf>}
:ANALysis:PMEAsure<x>:IDPRoximal:UNIT?
```
<x> = 1 to 2
<NRf>, <NRf>, <NRf> = See the DLM2000 Features Guide for this information.

Example
```
:ANALYSIS:PMEASURE1:IDPROXIMAL:UNIT -1,0,1
:ANALYSIS:PMEASURE1:IDPROXIMAL:UNIT?
-> :ANALYSIS:PMEASURE1:IDPROXIMAL:UNIT -1.0000000E+00,0.0000000E+00,1.0000000E+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 to 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 to 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 IAC
:ANALYSIS:PMEASURE1:INDICATOR?
-> :ANALYSIS:PMEASURE1:INDICATOR IAC
```

:ANALysis:PMEAsure<x>:MEASure?

Function Queries all the settings for automated measurement of power measurement parameters.

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

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

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

Function Queries the setting of a power measurement parameter.

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

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 to 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 a statistical value of a power measurement parameter.

Syntax
```
:ANALysis:PMEAsure<x>:MEASure:<Parameter>:{MAXimum|MEAN|MINimum|SDEViation}?
```
<x> = 1 to 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 Queries the on/off status of the power measurement parameter.

Syntax `:ANALysis:PMEAsure<x>:MEASure:<Parameter>:STATe {<Boolean>}`
`:ANALysis:PMEAsure<x>:MEASure:<Parameter>:STATe?`
<x> = 1 to 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 an automatically measured value of a power measurement parameter.

Syntax `:ANALysis:PMEAsure<x>:MEASure:<Parameter>:VALue?`
<x> = 1 to 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 parameter.

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

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

:ANALysis:PMEAsure<x>:MODE

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

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

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

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

Function Set 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 to 2
<Time> = −100.0 ns to 100.0 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|
C20|C50|C100|C200|C500|C1000|C2000}
:ANALysis:PMEAsure<x>:SETup:I:PROBe?
```
<x> = 1 to 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 to 2

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

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

Function Queries all voltage input channel settings for power measurement.

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

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

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

Function Set 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:DESKew?
```
<x> = 1 to 2
<Time> = −100.0 ns to 100.0 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 to 2
<NRf> = See the DLM2000 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 to 2
<NRf>, <NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:ANALYSIS:PMEASURE1:TRANGE 5,-5
:ANALYSIS:PMEASURE1:TRANGE?
-> :ANALYSIS:PMEASURE1:TRANGE 5.00000
00E+00,-5.0000000E+00
```

:ANALysis:PMEAsure<x>:UDPRoximal?

Function Queries all distal, mesial, and proximal settings.

Syntax
```
:ANALysis:PMEAsure<x>:UDPRoximal?
```
<x> = 1 to 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 to 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 to 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 to 2
<NRf>, <NRf>, <NRf> = See the DLM2000 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 to 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 to 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 to 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 to 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 to 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 to 2
<y> = 1 to 4

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

Description If the statistical value is immeasurable, the DLM2000 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 to 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>:STA
Te {<Boolean>}
:ANALysis:PMEAsure<x>:USER<y>:STATe?
```
<x> = 1 to 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>:UN
IT {<String>}
:ANALysis:PMEAsure<x>:USER<y>:UNIT?
```
<x> = 1 to 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>:VAL
ue? {<NRf>}
```
<x> = 1 to 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 DLM2000 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 DLM2000 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 set timeout.

Syntax
```
:ANALysis:WAIT? {<NRf>}
```
<NRf> = 1 to 36000 (the timeout specified in 100 ms intervals)

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

Description The command returns zero if the automated measurement finishes within the specified timeout. If automated measurement does not finish, or if it was never taking place to begin with, the command returns 1. Even if you set a long timeout, the command will return zero as soon as automated measurement finishes.

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 on/off status of auto calibration.

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 for 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,
  0V
  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 setting of a channel.

Syntax `:CHANnel<x>:BWIDth {FULL|<Frequency>}`
`:CHANnel<x>:BWIDth?`

<x> = 1 to 4 (1 or 2 for 2-channel models)

{<Frequency>} = See the DLM2000 Features Guide for this information.

Example
```
:CHANNEL1:BWIDTH FULL
:CHANNEL1:BWIDTH?
-> :CHANNEL1:BWIDTH FULL
```

:CHANnel<x>:COUPling

Function Sets or queries the input coupling setting of a channel.

Syntax `:CHANnel<x>:COUPling {AC|DC|DC50|GND}`
`:CHANnel<x>:COUPling?`

<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:CHANNEL1:COUPLING AC
:CHANNEL1:COUPLING?
-> :CHANNEL1:COUPLING AC
```

:CHANnel<x>:DESKew

Function Sets or queries the deskew setting of a channel.

Syntax `:CHANnel<x>:DESKew {<Time>}`
`:CHANnel<x>:DESKew?`

<x> = 1 to 4 (1 or 2 for 2-channel models)

<Time> = −100.0 ns to 100.0 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 status of a channel.

Syntax `:CHANnel<x>:DISPlay {<Boolean>}`
`:CHANnel<x>:DISPlay?`

<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:CHANNEL1:DISPLAY ON
:CHANNEL1:DISPLAY?
-> :CHANNEL1:DISPLAY 1
```

:CHANnel<x>:INVert

Function Sets or queries the on/off status of invert mode, which inverts the waveform display.

Syntax `:CHANnel<x>:INVert {<Boolean>}`
`:CHANnel<x>:INVert?`

<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:CHANNEL1:INVERT ON
:CHANNEL1:INVERT?
-> :CHANNEL1:INVERT 1
```

:CHANnel<x>:LABel?

Function Queries all waveform label settings of a channel.

Syntax `:CHANnel<x>:LABel?`

<x> = 1 to 4 (1 or 2 for 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 for 2-channel models)

<String> = Up to eight characters

Example
```
:CHANNEL1:LABEL:DEFINE "CH1"
:CHANNEL1:LABEL:DEFINE?
-> :CHANNEL1:LABEL:DEFINE "CH1"
```

Description You can only use the characters and symbols that appear on the DLM2000 soft keyboard.

:CHANnel<x>:LABel:DISPlay

Function Sets or queries the waveform label display on/off status of a channel.

Syntax
```
:CHANnel<x>:LABel:DISPlay {<Boolean>}
:CHANnel<x>:LABel:DISPlay?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:CHANNEL1:LABEL:DISPLAY ON
:CHANNEL1:LABEL:DISPLAY?
-> :CHANNEL1:LABEL:DISPLAY 1
```

:CHANnel<x>:LSCale?

Function Queries all linear scaling settings of a channel.

Syntax
```
:CHANnel<x>:LSCale?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

:CHANnel<x>:LSCale:AVALue

Function Sets or queries a scaling coefficient A value.

Syntax
```
:CHANnel<x>:LSCale:AVALue {<NRf>}
:CHANnel<x>:LSCale:AVALue?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:CHANNEL1:LSCALE:AVALUE 10
:CHANNEL1:LSCALE:AVALUE?
-> :CHANNEL1:LSCALE:AVALUE 10.0000E+0
0
```

:CHANnel<x>:LSCale:BVALue

Function Sets or queries an offset B value.

Syntax
```
:CHANnel<x>:LSCale:BVALue {<NRf>}
:CHANnel<x>:LSCale:BVALue?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:CHANNEL1:LSCALE:BVALUE 10
:CHANNEL1:LSCALE:BVALUE?
-> :CHANNEL1:LSCALE:BVALUE 10.0000E+0
0
```

:CHANnel<x>:LSCale:MODE

Function Sets or queries the on/off status of linear scaling.

Syntax
```
:CHANnel<x>:LSCale:MODE {<Boolean>}
:CHANnel<x>:LSCale:MODE?
```
<x> = 1 to 4 (1 or 2 for 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 for 2-channel models)
<String> = Up to four characters

Example
```
:CHANNEL1:LSCALE:UNIT "EU"
:CHANNEL1:LSCALE:UNIT?
-> :CHANNEL1:LSCALE:UNIT "EU"
```

Description You can only use the characters and symbols that appear on the DLM2000 soft keyboard. Assignment of a unit never affects the scale value.

:CHANnel<x>:OFFSet

Function Sets or queries the offset voltage of a channel.

Syntax
```
:CHANnel<x>:OFFSet {<Voltage>|<Curren
t>}:CHANnel<x>:OFFSet?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<Voltage>, <Current> = See the DLM2000 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 a channel.

Syntax
```
:CHANnel<x>:POSition {<NRf>}
:CHANnel<x>:POSition?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<NRf> = −4 to 4 (divisions)

Example
```
:CHANNEL1:POSITION 1
:CHANNEL1:POSITION?
-> :CHANNEL1:POSITION 1.00E+00
```

:CHANnel<x>:PROBe?

Function Queries all probe attenuation settings of a channel.

Syntax
```
:CHANnel<x>:PROBe?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

:CHANnel<x>:PROBe:DZCalibrate

Function Performs demagnetization and zero adjustment on each channel.

Syntax
```
:CHANnel<x>:PROBe:DZCalibrate
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:CHANNEL1:PROBE:DZCALIBRATE
```

:CHANnel<x>:PROBe[:MODE]

Function Sets or queries the probe attenuation setting of a 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 for 2-channel models)

Example
```
:CHANNEL1:PROBE:MODE <NRF>
:CHANNEL1:PROBE:MODE?
-> :CHANNEL1:PROBE:MODE <NRF>
```

:CHANnel<x>:PROBe:PZCalibrate

Function Executes zero calibration of current probes for a channel.

Syntax
```
:CHANnel<x>:PROBe:PZCalibrate
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:CHANNEL1:PROBE:PZCALIBRATE
```

:CHANnel<x>:VARiable

Function Sets or queries the voltage scale of a channel in 0.01 V/division steps.

Syntax
```
:CHANnel<x>:VARiable {<Voltage>|
<Current>}
:CHANnel<x>:VARiable?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<Voltage>, <Current> = See the DLM2000 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 voltage scale (in V/division) of a channel.

Syntax
```
:CHANnel<x>:VDIV {<Voltage>|
<Current>}
:CHANnel<x>:VDIV?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<Voltage>, <Current> = See the DLM2000 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 communication. 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 or not 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 or 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 information about the operation pending 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 remote or local. 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 (example:`CHANNEL1:PROBE:MODE 10`) or using abbreviation (example:`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
```

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 is incomplete.

When bit 6 (ACS) = 1:
Media access is incomplete.

When bit 8 (HST) = 1:
History searching is incomplete.

When bit 11 (SCH) = 1:
Edge or pattern searching is incomplete.

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 T-Y display.

Syntax `:CURSor:TY:DEGRee:ALL {<Boolean>}`

Example `:CURSOR:TY:DEGREE:ALL ON`

:CURSor[:TY]:DEGRee:D<x>?

Function Queries all angle cursor 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 status of an angle cursor 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 an angle cursor 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 of the settings for the angle difference between the angle cursors.

Syntax `:CURSor[:TY]:DEGRee:DD?`

:CURSor[:TY]:DEGRee:DD:STATe

Function Sets or queries the on/off status of the angle difference value ΔD, which is measured 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 value ΔD, which is measured 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 of the settings for the ΔV value, which is measured between the angle cursors.

Syntax `:CURSor[:TY]:DEGRee:DV?`

:CURSor[:TY]:DEGRee:DV:STATe

Function Sets or queries the on/off status of the ΔV value, which is measured 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, which is measured 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 Makes an angle cursor in the T-Y display jump 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 Jumps to the center of the zoom waveform. C1 and C2 are used to indicate Cursor1 and Cursor2.

:CURSor[:TY]:DEGRee:POSition<x>

Function Sets or queries the position of an angle cursor.

Syntax
```
:CURSor[:TY]:DEGRee:POSition<x> {<NRf
>}
:CURSor[:TY]:DEGRee:POSition<x>?
```
<x> = 1 or 2

<NRf> = −5 to 5 (in steps of 10 divisions/display record length)

Example
```
:CURSOR:TY:DEGREE:POSITION1 2
:CURSOR:TY:DEGREE:POSITION1?
-> :CURSOR:TY:DEGREE:POSITION1 2.0000
000
```

:CURSor[:TY]:DEGRee:REFerence<x>

Function Sets or queries the position of the angle reference start point (Reference1) or the angle reference end point (Reference2).

Syntax
```
:CURSor[:TY]:DEGRee:REFerence
<x> {<NRf>}
:CURSor[:TY]:DEGRee:REFerence<x>?
```
<x> = 1 or 2

<NRf> = −5 to 5 (in steps of 10 divisions/display record length)

Example
```
:CURSOR:TY:DEGREE:REFERENCE1 -1
:CURSOR:TY:DEGREE:REFERENCE1?
-> :CURSOR:TY:DEGREE:REFERENCE1 -1.00
00000
```

: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 that you want to measure using the angle cursors.

Syntax
```
:CURSor[:TY]:DEGRee:TRACe {<NRf>|ALL|
MATH<x>}
:CURSor[:TY]:DEGRee:TRACe?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

<x> = 1 or 2 (1 only for 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 unit of angle cursor measurement.

Syntax
```
:CURSor[:TY]:DEGRee:UNIT {<String>}
:CURSor[:TY]:DEGRee:UNIT?
```
<String> = Up to four 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 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 status of an angle cursor voltage.

Syntax
```
:CURSor[:TY]:DEGRee:V<x>:STATe {<Bool
ean>}
: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 an angle cursor voltage.

Syntax
```
:CURSor[:TY]:DEGRee:V<x>:VALue?
```
<x> = 1 or 2

Example
```
:CURSOR:TY:DEGREE:V1:VALUE?
-> :CURSOR:TY:DEGREE:V1:VALUE 10.0000
00E-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 T-Y display.

Syntax
```
:CURSor:TY:HORizontal:ALL {<Boolean>}
```
Example
```
:CURSOR:TY:HORIZONTAL:ALL ON
```

:CURSor[:TY]:HORizontal:DV?

Function Queries all of the settings for the vertical values between the ΔV cursors.

Syntax
```
:CURSor[:TY]:HORizontal:DV?
```

:CURSor[:TY]:HORizontal:DV:STATe

Function Sets or queries the on/off status of the vertical value measured between the ΔV cursors.

Syntax
```
:CURSor[:TY]:HORizontal:DV:STATe {<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 When linear scaling is on, the DLM2000 returns the scaled value.

:CURSor[:TY]:HORizontal:POSition<x>

Function Sets or queries the position of a vertical cursor.

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 The least significant digit for <NRf> is the hundredths place.

:CURSor[:TY]:HORizontal:TRACe

Function Sets or queries the source waveform that you want to measure using the ΔV cursors.

Syntax
```
:CURSor[:TY]:HORizontal:TRACe {<NRf>|MATH<x>}
:CURSor[:TY]:HORizontal:TRACe?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> = 1 or 2 (1 only for 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 vertical axis settings for a ΔV cursor.

Syntax
```
:CURSor[:TY]:HORizontal:V<x>?
```
<x> = 1 or 2

:CURSor[:TY]:HORizontal:V<x>:STATe

Function Sets or queries the on/off status of the vertical value measured for a ΔV cursor.

Syntax
```
:CURSor[:TY]:HORizontal:V<x>:STATe {<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 vertical value of a ΔV cursor.

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 When linear scaling is on, the DLM2000 returns the scaled value.

:CURSor[:TY]:MARKer?

Function Queries all marker cursor settings.

Syntax
```
:CURSor[:TY]:MARKer?
```

:CURSor[:TY]:MARKer:FORM

Function Sets or queries the marker cursor form.

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 for the specified marker.

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 T-Y 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<x>?

Function Queries all of the settings related to time measurement between marker cursors.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:DT<x>?
```
<x> = 1 to 4
<y> = 1 to 4

:CURSor[:TY]:MARKer:M<x>:DT<x>:STATe

Function Sets or queries the on/off status of the time value measured between marker cursors.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:DT<x>:STATe {<Boolean>}
:CURSor[:TY]:MARKer:M<x>:DT<x>: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<x>:VALue?

Function Queries the time value between marker cursors.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:DT<x>: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<x>?

Function Queries all of the settings related to vertical-axis measurement between marker cursors.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:DV<x>?
```
<x> = 1 to 4
<y> = 1 to 4

:CURSor[:TY]:MARKer:M<x>:DV<x>:STATe

Function Sets or queries the on/off status of the vertical value measured between marker cursors.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:DV<x>:STATe {<Boolean>}
:CURSor[:TY]:MARKer:M<x>:DV<x>: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 marker cursors.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:DV<x>: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 When linear scaling is on, the DLM2000 returns the scaled value.

:CURSor[:TY]:MARKer:M<x>:JUMP

Function Makes a marker cursor in the T-Y display jump 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 Jumps to the center of the zoom waveform.

:CURSor[:TY]:MARKer:M<x>:POSition

Function Sets or queries the time value of a marker cursor.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:POSition {<NRf>}
:CURSor[:TY]:MARKer:M<x>:POSition?
```
<x> = 1 to 4
<NRf> = −5 to 5 (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 time axis settings for a marker cursor.

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 status of the time value of a marker cursor.

Syntax
```
:CURSor[:TY]:MARKer:M<x>:T:STATe {<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 time value of a marker cursor.

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 that you want to measure using the marker cursors.

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 for 2-channel models)
<y> = 1 or 2 (1 only for 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 vertical axis settings for a marker cursor.

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 status of the vertical value of a marker cursor.

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 vertical value of a marker cursor.

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 1.500
0E-03
```

Description When linear scaling is on, the DLM2000 returns the scaled value.

:CURSor[:TY]:MARKer:FORM

Function Sets or queries the marker cursor form.

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]:TYPE

Function Sets or queries the cursor type.

Syntax
```
:CURSor[:TY]:TYPE {OFF|HORizontal|
HAVertical|MARKer|VERTical|DEGRee}
:CURSor[:TY]:TYPE?
```

Example
```
:CURSOR:TY:TYPE HORizontal
: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 T-Y display.

Syntax
```
:CURSor:TY:VERTical:ALL {<Boolean>}
```

Example
```
:CURSOR:TY:VERTICAL:ALL ON
```

:CURSor[:TY]:VERTical:DT?

Function Queries all time axis settings for the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DT?
```

:CURSor[:TY]:VERTical:DT:STATe

Function Sets or queries the on/off status of the time value measured between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DT:STATe {<Bool
ean>}
: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 vertical axis settings for the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DV?
```

:CURSor[:TY]:VERTical:DV:STATe

Function Sets or queries the on/off status of the vertical value measured between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:DV:STATe {<Bool
ean>}
: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 When linear scaling is on, the DLM2000 returns the scaled value.

:CURSor[:TY]:VERTical:JUMP

Function Makes a vertical cursor in the T-Y display jump 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 Jumps to the center of the zoom waveform. C1 and C2 are used to indicate V cursors Cursor1 and Cursor2.

:CURSor[:TY]:VERTical:PERDt?

Function Queries all time axis settings for 1/ΔT, which is measured between the ΔT cursors.

Syntax `:CURSor[:TY]:VERTical:PERDt?`

:CURSor[:TY]:VERTical:PERDt:STATe

Function Sets or queries the on/off status of the 1/ΔT, which is measured between the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:PERDt:STATe {<B
oolean>}
: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, which is measured 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 position of a ΔT cursor.

Syntax
```
:CURSor[:TY]:VERTical:POSition
<x> {<NRf>}
:CURSor[:TY]:VERTical:POSition<x>?
```
<x> = 1 or 2
<NRf> = −5 to 5 (in steps of 10 divisions/display record length)

Example
```
:CURSOR:TY:VERTICAL:POSITION1 2
:CURSOR:TY:VERTICAL:POSITION1?
-> :CURSOR:TY:VERTICAL:POSITION1 2.00
E+00
```

:CURSor[:TY]:VERTical:T<x>?

Function Queries all time axis settings for a ΔT cursor.

Syntax `:CURSor[:TY]:VERTical:T<x>?`
<x> = 1 or 2

:CURSor[:TY]:VERTical:T<x>:STATe

Function Sets or queries the on/off status of the time value of a ΔT cursor.

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 time value of a ΔT cursor.

Syntax `:CURSor[:TY]:VERTical:T<x>:VALue?`
<x> = 1 or 2

Example
```
:CURSOR:TY:VERTICAL:T1:VALUE?
-> :CURSOR:TY:VERTICAL:T1:VALUE -2.50
E-06
```

:CURSor[:TY]:VERTical:TRACe

Function Sets or queries the source waveform that you want to measure using the ΔT cursors.

Syntax
```
:CURSor[:TY]:VERTical:TRACe {<NRf>|AL
L|MATH<x>}
:CURSor[:TY]:VERTical:TRACe?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> = 1 or 2 (1 only for 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 vertical axis settings for a ΔT cursor.

Syntax `:CURSor[:TY]:VERTical:V<x>?`
<x> = 1 or 2

:CURSor[:TY]:VERTical:V<x>:STATe

Function Sets or queries the on/off status of the vertical value of a ΔT cursor.

Syntax
```
:CURSor[:TY]:VERTical:V<x>:STA
Te {<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 vertical value of a ΔT cursor.

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|P
ERSistence}
: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 {<Tim
e>|INFinity}
:DISPlay:ACCumulate:PERSistence?
```
<Time> = 100 ms to 100 s

Example
```
:DISPLAY:ACCUMULATE:PERSISTENCE 100ms
:DISPLAY:ACCUMULATE:PERSISTENCE?
-> :DISPLAY:ACCUMULATE:PERSISTENCE 10
0ms
```

: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|OR
ANge|PINK|PURPle|RED|SPINk|YELLow}
:DISPlay:COLor:{CHANnel<x>|MATH<x>}?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:DISPLAY:COLOR:CHANNEL1 BLUE
:DISPLAY:COLOR:CHANNEL1?
-> :DISPLAY:COLOR:CHANNEL1 BLUE
```

:DISPlay:COLor:LSTate

Function Sets or queries the color of the logic-waveform state display.

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|GREe
n|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:TRE
ND1 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}
:DISPlay:FORMat?
```

Example
```
:DISPLAY:FORMAT AUTO
:DISPLAY:FORMAT?
-> :DISPLAY:FORMAT AUTO
```

Description You cannot use {QUAD|HEXa} for the DLM2022, DLM2032, or DLM2052.

:DISPlay:FGRid

Function Sets or queries the on/off status of the fine grid display.

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 intensity settings for the 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|
MARKer|ZBOX} {<NRf>}
:DISPlay:INTENsity:{CURSor|GRID|
MARKer|ZBOX}?
```
<NRf> = 0 to 31

Example Below are examples for cursors.
```
: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 split screen mapping of a waveform.

Syntax
```
:DISPlay:SMAPping:CHANnel<x> {<NRf>}
:DISPlay:SMAPping:CHANnel<x>?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 4

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 of the settings for the FFT analysis feature.

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

:FFT<x>:AVERage?

Function Queries all FFT analysis averaging settings.

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 in 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 2^n 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

Description DL2000/DLM2000 firmware versions 2.10 and later support this command.

: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|MSBF
irst}
:FFT<x>:DATA:BYTeorder?
```
<x> = 1 or 2

Example
```
:FFT1:DATA:BYTEORDER LSBFIRST
:FFT1:DATA:BYTEORDER?
-> :FFT1:DATA:BYTEORDER LSBFIRST
```

Description DL2000/DLM2000 firmware versions 2.10 and later support this command.

: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
<NRf> = 0 to 125000

Example
```
:FFT1:DATA:END 125000
:FFT1:DATA:END?
-> :FFT1:DATA:END 125000
```

Description DL2000/DLM2000 firmware versions 2.10 and later support this command.

: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

Example
```
:FFT1:DATA:FORMAT ASCII
:FFT1:DATA:FORMAT?
-> :FFT:DATA:FORMAT ASCII
```

Description DL2000/DLM2000 firmware versions 2.10 and later support this command.

: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

Example
```
:FFT1:DATA:LENGTH?
-> :FFT1:DATA:LENGTH 6251
```

Description DL2000/DLM2000 firmware versions 2.10 and later support this command.

:FFT<x>:DATA:SEND?

Function Queries the FFT waveform data.

Syntax `:FFT<x>:DATA:SEND?`
<x> = 1 or 2

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.

Description DL2000/DLM2000 firmware versions 2.10 and later support this command.

: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
<NRf> = 0 to 125000

Example
```
:FFT1:DATA:START 0
:FFT1:DATA:START?
-> :FFT1:DATA:START 0
```

Description DL2000/DLM2000 firmware versions 2.10 and later support this command.

:FFT<x>:DISPlay

Function Sets or queries whether or not FFT analysis will 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 for FFT analysis.

Syntax
```
:FFT<x>:HORizontal?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:HORizontal:CSPan?

Function Queries all center and span settings for the horizontal axis for 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 for 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 62.5 GHz

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 value of the horizontal axis for 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 62.5 GHz

Example
```
:FFT1:HORIZONTAL:CSPAN:SPAN 1HZ
:FFT1:HORIZONTAL:CSPAN:SPAN?
-> :FFT1:HORIZONTAL:CSPAN:SPAN 1.000E+00
```

:FFT<x>:HORizontal:LRIGht?

Function Queries all of the setting for the left and right ends of the horizontal axis for 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 for the left and right ends of the horizontal axis for 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 62.5 GHz

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 FFT analysis horizontal axis mode.

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 for FFT analysis.

Syntax
```
:FFT<x>:LENGth {<NRf>}
:FFT<x>:LENGth?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1250, 12500, 125000, 250000

Example
```
:FFT1:LENGTH 1250
:FFT1:LENGTH?
-> :FFT1:LENGTH 1250
```

:FFT<x>:MEASure?

Function Queries all automatic measurement settings for FFT analysis.

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

:FFT<x>:MEASure:MARKer?

Function Queries all marker cursor measurement settings for 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 Basic marker cursor item settings for 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 Basic marker cursor items for 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 of the settings for frequency measurement between marker cursors for 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 status of the frequency measured between marker cursors for 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 for 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 of the settings for level measurement between marker cursors for 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 status of the level measured between marker cursors for 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 for 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 of the settings for frequency measurement of a marker cursor for 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 status of the frequency of marker cursor for FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:FREQue
ncy<y>:STATe {<Boolean>}
:FFT<x>:MEASure:MARKer[:BASic]:FREQue
ncy<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:FREQUEN
CY1:STATE 1
```

:FFT<x>:MEASure:MARKer[:BASic]:FREQuency<y>:VALue?

Function Queries the frequency of a marker cursor for FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:FREQue
ncy<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:FREQUEN
CY1:VALUE 1.000E+00
```

:FFT<x>:MEASure:MARKer[:BASic]:POSition<y>

Function Sets or queries the position of a marker cursor for FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:POSiti
on<y> {<NRf>}
:FFT<x>:MEASure:MARKer[:BASic]:POSiti
on<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = −5 to 5 (divisions)

Example
```
:FFT1:MEASURE:MARKER:BASIC:POSITI
ON1 1
:FFT1:MEASURE:MARKER:BASIC:POSITION1?
-> :FFT1:MEASURE:MARKER:BASIC:POSITI
ON1 1.000E+00
```

:FFT<x>:MEASure:MARKer[:BASic]:V<y>?

Function Queries all of the settings for level measurement of a marker cursor for 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 status of the level of marker cursor for FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:S
TATe {<Boolean>}
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:S
TATe?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:MARKER:BASIC:V1:STA
TE ON
:FFT1:MEASURE:MARKER:BASIC:V1:STATE?
-> :FFT1:MEASURE:MARKER:BASIC:V1:STA
TE 1
```

:FFT<x>:MEASure:MARKer[:BASic]:V<y>:VALue?

Function Queries the level of a marker cursor for FFT analysis.

Syntax
```
:FFT<x>:MEASure:MARKer[:BASic]:V<y>:V
ALue?
```
<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:VAL
UE 1.000E+00
```

:FFT<x>:MEASure:MODE

Function Sets or queries the FFT analysis automatic measurement mode.

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 peak measurement settings for FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:PEAK[:BASic]?

Function Queries all Basic peak item settings for FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK:BASic?
```
<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:PEAK[:BASic]:ALL

Function Collectively turns on or off all Basic peak items for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:ALL {<Boolean>}`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:MEASURE:PEAK:BASIC:ALL ON`

:FFT<x>:MEASure:PEAK[:BASic]:DFRequency?

Function Queries all of the settings for frequency measurement between peak values for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:DFRequency?`

<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:PEAK[:BASic]:DFRequency:STATe

Function Sets or queries the on/off status of the frequency measured between peak values for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:DFRequency:STATe {<Boolean>}`

`:FFT<x>:MEASure:PEAK[:BASic]:DFRequency:STATe?`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:MEASURE:PEAK:BASIC:DFREQUENCY:STATE ON`

`:FFT1:MEASURE:PEAK:BASIC:DFREQUENCY:STATE?`

`-> :FFT1:MEASURE:PEAK:BASIC:DFREQUENCY:STATE 1`

:FFT<x>:MEASure:PEAK[:BASic]:DFRequency:VALue?

Function Queries the frequency between peak values for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:DFRequency:VALue?`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:MEASURE:PEAK:BASIC:DFREQUENCY:VALUE?`

`-> :FFT1:MEASURE:PEAK:BASIC:DFREQUENCY:VALUE 1.000E+00`

:FFT<x>:MEASure:PEAK[:BASic]:DV?

Function Queries all of the settings for power measurement between peak values for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:DV?`

<x> = 1 or 2 (1 only for 2-channel models)

:FFT<x>:MEASure:PEAK[:BASic]:DV:STATe

Function Sets or queries the on/off status of the power measured between peak values for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:DV:STATe {<Boolean>}`

`:FFT<x>:MEASure:PEAK[:BASic]:DV:STATe?`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:MEASURE:PEAK:BASIC:DV:STATE ON`

`:FFT1:MEASURE:PEAK:BASIC:DV:STATE?`

`-> :FFT1:MEASURE:PEAK:BASIC:DV:STATE 1`

:FFT<x>:MEASure:PEAK[:BASic]:DV:VALue?

Function Queries the power between peak values for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:DV:VALue?`

<x> = 1 or 2 (1 only for 2-channel models)

Example `:FFT1:MEASURE:PEAK:BASIC:DV:VALUE?`

`-> :FFT1:MEASURE:PEAK:BASIC:DV:VALUE 1.000E+00`

:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>?

Function Queries all of the settings for a peak frequency for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>?`

<x> = 1 or 2 (1 only for 2-channel models)

<y> = 1 or 2

:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>:STATe

Function Sets or queries the on/off status of a peak frequency value for FFT analysis.

Syntax `:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>:STATe {<Boolean>}`

`:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>:STATe?`

<x> = 1 or 2 (1 only for 2-channel models)

<y> = 1 or 2

Example `:FFT1:MEASURE:PEAK:BASIC:FREQUENCY1:STATE ON`

`:FFT1:MEASURE:PEAK:BASIC:FREQUENCY1:STATE?`

`-> :FFT1:MEASURE:PEAK:BASIC:FREQUENCY1:STATE 1`

:FFT<x>:MEASure:PEAK[:BASic]:FREQuency<y>:VALue?

Function Queries a peak frequency value for FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK[:BASic]:FREQuenc
y<y>:VALue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:PEAK:BASIC:FREQUENCY1:V
ALUE?
-> :FFT1:MEASURE:PEAK:BASIC:FREQUENCY
1:VALUE 1.000E+00
```

:FFT<x>:MEASure:PEAK[:BASic]:RANGe<y>

Function Sets or queries the measurement range of a peak value for FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK[:BASic]:RANGe
<y> {<NRf>}
:FFT<x>:MEASure:PEAK[:BASic]:RANGe
<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<NRf> = −5 to 5 (divisions)

Example
```
:FFT1:MEASURE:PEAK:BASIC:RANGE1 0,1
:FFT1:MEASURE:PEAK:BASIC:RANGE1?
-> :FFT1:MEASURE:PEAK:BASIC:RANGE1 1.
000E+00,0.000E+00
```

:FFT<x>:MEASure:PEAK[:BASic]:V<y>?

Function Queries all of the settings for a peak value for FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK[:BASic]:V<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

:FFT<x>:MEASure:PEAK[:BASic]:V<y>:STATe

Function Sets or queries the on/off status of a peak value for FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK[:BASic]:V<y>:STA
Te {<Boolean>}
:FFT<x>:MEASure:PEAK[:BASic]:V<y>:STA
Te?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:PEAK:BASIC:V1:STATE ON
:FFT1:MEASURE:PEAK:BASIC:V1:STATE?
-> :FFT1:MEASURE:PEAK:BASIC:V1:
STATE 1
```

:FFT<x>:MEASure:PEAK[:BASic]:V<y>:VALue?

Function Queries a peak value for FFT analysis.

Syntax
```
:FFT<x>:MEASure:PEAK[:BASic]:V<y>:VAL
ue?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:FFT1:MEASURE:PEAK:BASIC:V1:VALUE?
-> :FFT1:MEASURE:PEAK:BASIC:V1:
VALUE 1.000E+00
```

:FFT<x>:MODE

Function Sets or queries the waveform display mode for 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 for 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 point for magnifying the vertical axis for FFT analysis.

Syntax
```
:FFT<x>:RPOSition {<NRf>}
:FFT<x>:RPOSition?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = −4 to 4 (divisions)

Example
```
:FFT1:RPOSITION 1
:FFT1:RPOSITION?
-> :FFT1:RPOSITION 1.000E+00
```

:FFT<x>:TRACe

Function Sets or queries the source waveform for FFT analysis.

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

Example
```
:FFT1:TRACE 1
:FFT1:TRACE?
-> :FFT1:TRACE 1
```

:FFT<x>:TYPE

Function Sets or queries the spectrum for 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> of FFT<x> = 1 or 2 (1 only for 2-channel models)

<NRf> = 1 to 4 (1 or 2 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

Example
```
:FFT1:TYPE CH_MAG,1
:FFT1:TYPE?
-> :FFT1:TYPE CH_MAG,1
```

Description This command results in an error on models that are not equipped with the user-defined computation option.

- 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), use <NRf> to select the second waveform under analysis.

:FFT<x>:UNIT?

Function Queries all FFT computation unit settings.

Syntax
```
:FFT<x>:UNIT?
```
<x>= 1 or 2(1 only for the DLM2022, DLM2032, DLM2052)

:FFT<x>:UNIT[:DEFine]

Function Sets or queries an FFT computation unit.

Syntax
```
:FFT<x>:UNIT[:DEFine] {<String>}
:FFT<x>:UNIT[:DEFine]?
```
<x>= 1 or 2(1 only for the DLM2022, DLM2032, DLM2052)

Example
```
:FFT1:UNIT:DEFINE "EU"
:FFT1:UNIT:DEFINE?
-> :FFT1:UNIT:DEFINE "EU"
```

Description Units are applied to scale values. They never affect 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 the DLM2022, DLM2032, DLM2052)

Description
```
:FFT1:UNIT:MODE AUTO
:FFT1:UNIT:MODE?
-> :FFT1:UNIT:MODE AUTO
```

:FFT<x>:VERTical?

Function Queries all vertical axis settings for 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 for 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 FFT analysis vertical axis mode.

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 sensitivity for FFT analysis.

Syntax
```
:FFT<x>:VERTical:SENSitivity {<NRf>}
:FFT<x>:VERTical:SENSitivity?
```
<x> = 1 or 2 (1 only for 2-channel models)

<NRf> = −1.0000E+31 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 status of the VT waveform display for 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 for 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 settings for data storage.

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 DLM2000 Features Guide for this information.

Example `:FILE:COPY:CDIRECTORY "UTIL"`

(relative path designation)

:FILE:COPY:DRIVe

Function Sets the file copy source medium.

Syntax `:FILE:COPY:DRIVe {FLAShmem|USB,<NRf>}`

<NRf> = 0 to 3

Example `:FILE:COPY:DRIVE FLASHMEM`

Description You can omit the <NRf> for USB if the drive is not partitioned or is not divided into LUNs.

:FILE:COPY[:EXECute]

Function Executes file copying. This is an overlap command.

Syntax `:FILE:COPY[:EXECute] {<String>}`

<String> = See the DLM2000 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 a specific type of data file. This is an overlap command.

Syntax
```
:FILE:DELete:{AHIStogram|ASCii|
BINary|BMP|FFT|HLISt|JPEG|MEASure|
PNG|SBUS|SETup|SNAP|ZPOLygon|ZWAVe}
[:EXECute] {<File name>}
```

Example Below is an example for waveform data.
```
:FILE:DELETE:AHISTOGRAM:
EXECUTE "DATA"
```

Description Use the `:FILEDIRectory:DRIVe` command to select the target medium.

:FILE[:DIRectory]:CDIRectory

Function Changes the current directory.

Syntax
```
:FILE[:DIRectory]:
CDIRectory {<String>}
```

<String> =See the DLM2000 Features Guide for this information.

Example `:FILE:DIRECTORY:CDIRECTORY "UTIL"`

:FILE[:DIRectory]:DRIVe

Function Sets the current medium.

Syntax
```
:FILE[:DIRectory]:DRIVe {FLAShmem|
NETWork|USB,<NRf>}
```

<NRf> = 0 to 3

Example `:FILE:DIRECTORY:DRIVE FLASHMEM`

Description You can omit the <NRf> for USB if the drive is not partitioned or is not divided into LUNs.

:FILE[:DIRectory]:FREE?

Function Queries the free space on the current medium in bytes.

Syntax `:FILE[:DIRectory]:FREE?`

Example
```
:FILE:DIRECTORY:FREE?
-> :FILE:DIRECTORY:FREE 65536
```

:FILE[:DIRectory]:MDIRectory

Function Creates a directory in the current directory. This is an overlap command.

Syntax
```
:FILE[:DIRectory]:MDIRectory {<String
>}
```

<String> = See the DLM2000 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 Loads waveform data. This is an overlap command.

Syntax `:FILE:LOAD:BINary[:EXECute] {<File name>{,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 a specific type of data.

Syntax `:FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}:ABORt`
<x> = 1 to 4

Example Below is an example for setup data.
`:FILE:LOAD:SETUP:ABORT`

:FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}[:EXECute]

Function Loads a specific type of data. This is an overlap command.

Syntax `:FILE:LOAD:{SETup|SNAP|ZPOLygon<x>|ZWAVe<x>}[:EXECute] {<File name>}`
<x> = 1 to 4

Example Below is an example 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 DLM2000 Features Guide for this information.

Example `:FILE:MOVE:CDIRECTORY "UTIL"`

:FILE:MOVE:DRIVe

Function Sets the file move destination medium.

Syntax `:FILE:MOVE:DRIVe {FLAShmem|USB,<NRf>}`
<NRf> = 0 to 3

Example `:FILE:MOVE:DRIVE FLASHMEM`

Description You can omit the <NRf> for USB if the drive is not partitioned or is not divided into LUNs.

:FILE:MOVE[:EXECute]

Function Executes file moving. This is an overlap command.

Syntax `:FILE:MOVE[:EXECute] {<String>}`
<String> = See the DLM2000 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 file protection on or off.

Syntax `:FILE:PROTect[:EXECute] {<String>,<Boolean>}`
<String> = See the DLM2000 Features Guide for this information.

Example `:FILE:PROTECT:EXECUTE "DATA.PNG",ON`

:FILE:REName[:EXECute]

Function Renames a file.

Syntax `:FILE:REName[:EXECute] {<String>,<String>}`
<String> = See the DLM2000 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 of the settings related to the saving of a specified 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 data.

Syntax
```
:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}:ABORt
```

Example Below is an example for waveform data.
```
:FILE:SAVE:BINARY:ABORT
```

:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HLISt|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}[:EXECute]

Function Saves a specific type of file. This is an overlap command.

Syntax
```
:FILE:SAVE:{AHIStogram|ASCii|BINary|FFT|HARMonics|MEASure|SBUS|SETup|SNAP|ZWAVe}[:EXECute] {<File name>}
```

Example Below is an example for waveform data.
```
:FILE:SAVE:BINARY:EXECUTE "DATA"
```

:FILE:SAVE:{AHIStogram|FFT|HARMonics|SBUS|ZWAVe}:SELect

Function Sets or queries the area of a specific type of data to save.

Syntax
```
:FILE:SAVE:AHIStogram:SELect {<NRf>}
:FILE:SAVE:AHIStogram:SELect?
```
<NRf> = 1 or 2 (1 to 4 for ZWAVe)

Example Below are examples 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 status of the auto naming feature for saving files.

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 of a specific type of data.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:COMPression {DECimation|OFF|PTOPeak}
:FILE:SAVE:{ASCii|BINary}:COMPression?
```

Example Below is an example 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 how history waveforms will be saved.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:HISTory {ALL|AVERage|ONE}
:FILE:SAVE:{ASCii|BINary}:HISTory?
```

Example Below is an example 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}:LENGth {<NRf>}
:FILE:SAVE:{ASCii|BINary}:LENGth?
```
<NRf> = See the DLM2000 Features Guide for this information.

Example Below is an example for waveform data.
```
:FILE:SAVE:BINARY:LENGTH
:FILE:SAVE:BINARY:LENGTH?
-> :FILE:SAVE:BINARY:LENGTH 1250
```

:FILE:SAVE:{ASCii|BINary}:RANGe

Function Sets or queries the save range for a specific type of data.

Syntax
```
:FILE:SAVE:{ASCii|BINary}:RANGe {MAIN|Z1|Z2}
:FILE:SAVE:{ASCii|BINary}:RANGe?
```

Example Below is an example 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 waveforms to save for 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 for 2-channel models)
<x> = 1 or 2 (1 only for 2-channel models)

Example Below is an example for waveform data.
```
: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 {<Boole
an>}
:FILE:SAVE:ASCii:TINFormation?
```
Example
```
:FILE:SAVE:ASCII:TINFORMATION ON
:FILE:SAVE:ASCII:TINFORMATION?
-> :FILE:SAVE:ASCII:TINFORMATION 1
```

:FILE:SAVE:FFT:FINFormation (Frequency Information)

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:COMMent

Function Sets or queries the comment that will be attached to the data that will 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 that appear on the DLM2000 soft keyboard.

:FILE:SAVE:NAME

Function Sets or queries the file name for the data that will be saved.

Syntax
```
:FILE:SAVE:NAME {<File name>}
:FILE:SAVE:NAME?
```
Example
```
:FILE:SAVE:NAME "CASE1"
:FILE:SAVE:NAME?
-> :FILE:SAVE:NAME "CASE1"
```

:FILE:SAVE:SBUS:COMPression

Function Sets or queries the save compression method of SENT data.

Syntax
```
:FILE:SAVE:SBUS:COMPression {DECimati
on|OFF|PTOPeak}
:FILE:SAVE:SBUS:COMPression?
```
Example
```
:FILE:SAVE:SBUS:COMPRESSION DECIMATIO
N
:FILE:SAVE:SBUS:COMPRESSION?
-> :FILE:SAVE:SBUS:COMPRESSION DECIMA
TION
```

: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
```

:FILE:SAVE:SBUS:LENGth

Function Sets or queries the number of data points to save when compressing or sampling (decimating) SENT data.

Syntax
```
:FILE:SAVE:SBUS:LENGth {<NRf>}
:FILE:SAVE:SBUS:LENGth?
```
<NRf> = 1250, 12500, 125000, 1250000

Example
```
:FILE:SAVE:SBUS:LENGTH 1250
:FILE:SAVE:SBUS:LENGTH?
-> :FILE:SAVE:SBUS:LENGTH 1250
```

:FILE:SAVE:SBUS:TINFormation (Time Information)

Function Sets or queries whether SENT waveform data is saved with time information (ON) or without it (OFF).

Syntax
```
:FILE:SAVE:SBUS:TINFormation {<Boolea
n>}
:FILE:SAVE:SBUS:TINFormation?
```
Example
```
:FILE:SAVE:SBUS:TINFORMATION ON
:FILE:SAVE:SBUS:TINFORMATION?
-> :FILE:SAVE:SBUS:TINFORMATION 1
```

:FILE:SAVE:SBUS:TYPe

Function Sets or queries the save method of SENT data.

Syntax
```
:FILE:SAVE:SBUS:TYPe {LIST|
TWAVeform}
:FILE:SAVE:SBUS:TYPe?
```
Example
```
:FILE:SAVE:SBUS:TYPE LIST
:FILE:SAVE:SBUS:TYPE?
-> :FILE:SAVE:SBUS:TYPE LIST
```

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 of the settings related to the action executed when GO/NO-GO results are NO-GO and the reference.

Syntax `:GONogo:ACTion?`

:GONogo:ACTion:BUZZer

Function Sets or queries whether or not the DLM2000 will sound an alarm when a GO/NO-GO result 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 or not the DLM2000 will print screen images to the optional built-in printer or an external printer for NO-GO results.

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 of the settings related to email notification for NO-GO results.

Syntax `:GONogo:ACTion:MAIL?`

Description You can use this command when the optional Ethernet interface is installed.

:GONogo:ACTion:MAIL:COUNt

Function Sets or queries the upper limit of emails that will be sent for NO-GO results.

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
```

Description You can use this command when the optional Ethernet interface is installed.

:GONogo:ACTion:MAIL:MODE

Function Sets or queries whether or not the DLM2000 will send email notification for NO-GO results.

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
```

Description You can use this command when the optional Ethernet interface is installed.

:GONogo:ACTion:SAVE

Function Sets or queries whether or not the DLM2000 will save waveform data to the storage medium for NO-GO results.

Syntax
```
:GONogo:ACTion:SAVE {<Boolean>}
:GONogo:ACTion:SAVE?
```

Example
```
:GONOGO:ACTION:SAVE ON
:GONOGO:ACTION:SAVE?
-> :GONOGO:ACTION:SAVE 1
```

Description Set or query the media type by using the `:FILE:DIRECTORY:DRIVE`.

:GONogo:COUNt?

Function Queries the number of performed GO/NO-GO determinations.

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 logic.

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 at which the DLM2000 will stop performing determinations.

Syntax
```
:GONogo:NGStopcount {<NRf>|UNFinity}
:GONogo:NGStopcount?
```
<NRf> = 1 to 1000

Example
```
:GONOGO:NGSTOPCOUNT 100
:GONOGO:NGSTOPCOUNT?
-> :GONOGO:NGSTOPCOUNT 100
```

:GONogo:STOPcount

Function Sets or queries the number of acquisitions at which the DLM2000 will stop performing determinations.

Syntax
```
:GONogo:STOPcount {<NRf>|UNFinity}
: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 by using a timeout timer.

Syntax
```
:GONogo:WAIT? {<NRf>}
```
<NRf> = 0 to 864000 (in 100-ms steps)

Example (Set the timeout to 5 seconds)
```
:GONOGO:WAIT? 50
-> :GONOGO:WAIT 0
```

Description The DLM2000 returns 0 if the operation finishes before the timer expires and returns 1 if the timer expires.

:GONogo[:ZPARameter]?

Function Queries all zone and parameter determination settings.

Syntax
```
:GONogo[:ZPARameter]?
```

:GONogo[:ZPARameter]:NUMBer<x>?

Function Queries all of the settings related to the specified reference standard for zone or parameter determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>?
```
<x> = 1 to 4

:GONogo[:ZPARameter]:NUMBer<x>:CAUSe?

Function Queries whether or not the specified waveform parameter is the cause of a NO-GO zone or parameter determination.

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 DLM2000 returns 1. Otherwise, the DLM2000 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>:CONDit
ion {IN|OFF|OUT}
:GONogo[:ZPARameter]:NUMBer<x>:CONDit
ion?
```
<x> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:CONDITI
ON IN
:GONOGO:ZPARAMETER:NUMBER1:CONDITION?
-> :GONOGO:ZPARAMETER:NUMBER1:CONDITI
ON IN
```

:GONogo[:ZPARameter]:NUMBer<x>:MODE

Function Sets or queries the specified reference standard mode.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:MO
DE {PARameter|POLYgon|RECTangle|
WAVE}
:GONogo[:ZPARameter]:NUMBer<x>:MODE?
```
<x> = 1 to 4

Example
```
:GONOGO:ZPARAMETER:NUMBER1:MODE PARAM
ETER
:GONOGO:ZPARAMETER:NUMBER1:MODE?
-> :GONOGO:ZPARAMETER:NUMBER1:MODE PA
RAMETER
```

:GONogo[:ZPARameter]:NUMBer<x>:PARameter?

Function Queries all parameter settings for the specified reference standard.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:PARame
ter?
```
<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 DLM2000 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 for 2-channel models)

<y> of BIT<y> = 1 to 8

<y> of FFT<y> = 1 or 2 (1 only for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

<y> of XY<y> = 1 or 2 (1 only for 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 specified waveform parameter’s measured value 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 When the mode is set to OFF or when the value is otherwise immeasurable, the DLM2000 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 that will be used for polygonal zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:HPOSition {<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:HPOSition?
```
<x> = 1 to 4
<NRf> = −5 to 5 (divisions)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:HPOSITION 1
:GONOGO:ZPARAMETER:NUMBER1:POLYGON:HPOSITION?
-> :GONOGO:ZPARAMETER:NUMBER1:
POLYGON:HPOSITION 1.000E+00
```

Description When TRACe is set to XY<x>, the <NRf> is 1 to 4 (div).

:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:RANGe

Function Sets or queries the source window that will be used for 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 that will be used for 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 for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)
<y> of XY<y> = 1 or 2 (1 only for 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 that will be used for polygonal zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:VPOSition {<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:POLYgon:VPOSition?
```
<x> = 1 to 4
<NRf> = −4 to 4 (divisions)

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 that will be used for 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 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 that will be used for rectangular zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:HORizontal {<NRf>,<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:HORizontal?
```
<x> = 1 to 4
<NRf> = −5 to 5 (divisions)

Example
```
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:HORIZONTAL 1,2
:GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:HORIZONTAL?
-> :GONOGO:ZPARAMETER:NUMBER1:RECTANGLE:HORIZONTAL 2.000E+00,1.000E+00
```

:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:RANGe

Function Sets or queries the source window of the rectangle that will be used for 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
```

Description When TRACe is set to XY<x>, the <NRf> is 1 to 4 (div).

:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:TRACe

Function Sets or queries the source waveform of the rectangle that will be used for 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 for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)
<y> of XY<y> = 1 or 2 (1 only for 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 that will be used for rectangular zone determination.

Syntax
```
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:VERTical {<NRf>,<NRf>}
:GONogo[:ZPARameter]:NUMBer<x>:RECTangle:VERTical?
```
<x> = 1 to 4
<NRf> = −4 to 4 (divisions)

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 or 2 (1 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 Executes whole editing 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 that will be used for 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 that will be used for 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 for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 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 area that will be used for waveform zone determination.

Syntax `:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRANge {<NRf>,<NRf>}`
`:GONogo[:ZPARameter]:NUMBer<x>:WAVE:TRANge?`
<NRf> = −5 to 5 (divisions)

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 that will be used for 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:COMMent

Function Sets or queries the comment at the lower right of the screen.

Syntax
```
:HCOPy:COMMent {<String>}
:HCOPy:COMMent?
```
<String> = Up to 32 characters

Example
```
:HCOPY:COMMENT "THIS IS TEST."
:HCOPY:COMMENT?
-> :HCOPY:COMMENT "THIS IS TEST."
```

:HCOPy:DIRection

Function Sets or queries the data output destination.

Syntax
```
:HCOPy:DIRection {EXTPrinter|PRINter|NETPrinter|FILE|MULTitarget}
:HCOPy:DIRection?
```

Example
```
:HCOPY:DIRECTION EXTPRINTER
:HCOPY:DIRECTION?
-> :HCOPY:DIRECTION EXTPRINTER
```

Description You can use “NETPrint” when the optional Ethernet interface is installed.

: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 whether the DLM2000 will print to the external printer in normal copy or hard copy mode.

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 setting.

Syntax
```
:HCOPy:EXTPrinter:TONE {<Boolean>}
:HCOPy:EXTPrinter:TONE?
```

Example
```
:HCOPY:EXTPRINTER:TONE ON
:HCOPY:EXTPRINTER:TONE?
-> :HCOPY:EXTPRINTER:TONE 1
```

:HCOPy:EXTPrinter:TYPE

Function Sets or queries the external printer output command type.

Syntax
```
:HCOPy:EXTPrinter:TYPE {EINKjet|HINKjet}
:HCOPy:EXTPrinter:TYPE?
```

Example
```
:HCOPY:EXTPRINTER:TYPE EINKJET
:HCOPY:EXTPRINTER:TYPE?
-> :HCOPY:EXTPRINTER:TYPE EINKJET
```

: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
```

Description You can use this command when the optional Ethernet interface is installed.

: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 {<Boolean
>}
: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?
```

Description You can use this command when the optional Ethernet interface is installed.

:HCOPy:NETPrint:MODE

Function Sets or queries whether the DLM2000 will print to the network printer in normal copy or hard copy mode.

Syntax
```
:HCOPy:NETPrint:MODE {HARD|NORMal}
:HCOPy:NETPrint:MODE?
```

Example
```
:HCOPY:NETPRINT:MODE HARD
:HCOPY:NETPRINT:MODE?
-> :HCOPY:NETPRINT:MODE HARD
```

Description You can use this command when the optional Ethernet interface is installed.

:HCOPy:NETPrint:TONE

Function Sets or queries the network printer output color setting.

Syntax
```
:HCOPy:NETPrint:TONE {<Boolean>}
:HCOPy:NETPrint:TONE?
```

Example
```
:HCOPY:NETPRINT:TONE ON
:HCOPY:NETPRINT:TONE?
-> :HCOPY:NETPRINT:TONE 1
```

Description You can use this command when the optional Ethernet interface is installed.

:HCOPy:NETPrint:TYPE

Function Sets or queries the network printer output command type.

Syntax
```
:HCOPy:NETPrint:TYPE {EINKjet|HINKjet
|HLASer}
:HCOPy:NETPrint:TYPE?
```

Example
```
:HCOPY:NETPRINT:TYPE EINKJET
:HCOPY:NETPRINT:TYPE?
-> :HCOPY:NETPRINT:TYPE EINKJET
```

Description You can use this command when the optional Ethernet interface is installed.

:HCOPy:PRINter?

Function Queries all built-in printer output settings.

Syntax
```
:HCOPy:PRINter?
```

:HCOPy:PRINter:MAG

Function Sets or queries the magnification used when the DLM2000 prints to the built-in printer in long copy mode.

Syntax
```
:HCOPy:PRINter:MAG {<NRf>}
:HCOPy:PRINter:MAG?
```
<NRf> = 2 to 10

Example
```
:HCOPY:PRINTER:MAG 2
:HCOPY:PRINTER:MAG?
-> :HCOPY:PRINTER:MAG 2.000E+00
```

:HCOPy:PRINter:MODE

Function Sets or queries whether the DLM2000 will print to the built-in printer in short, long, or hard copy mode.

Syntax
```
:HCOPy:PRINter:MODE {HARD|LONG|SHORt}
:HCOPy:PRINter:MODE?
```

Example
```
:HCOPY:PRINTER:MODE HARD
:HCOPY:PRINTER:MODE?
-> :HCOPY:PRINTER:MODE HARD
```

:HCOPy:PRINter:RANGe

Function Sets or queries the source window used when the DLM2000 prints to the built-in printer in long copy mode.

Syntax
```
:HCOPy:PRINter:RANGe {MAIN|Z1|Z2}
:HCOPy:PRINter:RANGe?
```

Example
```
:HCOPY:PRINTER:RANGE MAIN
:HCOPY:PRINTER:RANGE?
-> :HCOPY:PRINTER:RANGE MAIN
```

:HCOPy:PRINter:REPort

Function Sets or queries whether the DLM2000 will print additional information on the built-in printer.

Syntax
```
:HCOPy:PRINter:REPort {<Boolean>}
:HCOPy:PRINter:REPort?
```

Example
```
:HCOPY:PRINTER:REPORT ON
:HCOPY:PRINTER:REPORT?
-> :HCOPY:PRINTER:REPORT 1
```

:HCOPy:PRINter:TRANge

Function Sets or queries the output range used when the DLM2000 prints to the built-in printer in long copy mode.

Syntax
```
:HCOPy:PRINter:TRANge {<NRf>,<NRf>}
:HCOPy:PRINter:TRANge?
```
<NRf> = −5 to 5 (in steps of 10 divisions/display record length)

Example
```
:HCOPY:PRINTER:TRANGE −5
:HCOPY:PRINTER:TRANGE?
-> :HCOPY:PRINTER:TRANGE −5
```

5.15 HISTory Group

:HISTory?

Function Queries all of the settings for the history feature.

Syntax `:HISTory?`

:HISTory:AVERage

Function Sets or queries the highlight display mode for history waveforms.

Syntax
```
:HISTory:AVERage {<Boolean>}
:HISTory:AVERage?
```

Example
```
:HISTORY:AVERAGE ON
:HISTORY:AVERAGE?
-> :HISTORY:AVERAGE 1
```

:HISTory:DISPlay

Function Sets or queries the display record start and end numbers.

Syntax
```
:HISTory:DISPlay {<NRf>,<NRf>}
:HISTory:DISPlay?
```
<NRf> = See the DLM2000 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 DLM2000 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 minimum history waveform record number.

Syntax `:HISTory:RECord? {MINimum}`

Example
```
:HISTORY:RECORD? MINIMUM
-> :HISTORY:RECORD -1
```

:HISTory:REPLay?

Function Queries all of the settings for the history waveform replay feature.

Syntax `:HISTory:REPLay?`

:HISTory:REPLay:JUMP

Function Jumps to the specified record number in a 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, or 10

Example
```
:HISTORY:REPLAY:SPEED 1
:HISTORY:REPLAY:SPEED?
-> :HISTORY:REPLAY:SPEED 1
```

:HISTory:REPLay:STARt

Function Starts replaying a history waveform in the specified direction.

Syntax
```
:HISTory:REPLay:STARt {MAXimum|
MINimum}
```

Example `:HISTORY:REPLAY:START MAXIMUM`

:HISTory:REPLay:STOP

Function Stops history waveform replaying.

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 searching.

Syntax `:HISTory[:SEARch]:ABORt`

Example `:HISTORY:SEARCH:ABORT`

:HISTory[:SEARch]:EXECute

Function Executes searching. 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]:RESet

Function Resets history waveform search conditions.

Syntax `:HISTory[:SEARch]:RESet`

Example `:HISTORY:SEARCH:RESET`

:HISTory[:SEARch]:NUMBer<x>?

Function Queries all of the settings for a search condition.

Syntax `:HISTory[:SEARch]:NUMBer<x>?`

<x> = 1 to 4

:HISTory[:SEARch]:NUMBer<x>:CONDition

Function Sets or queries the search criterion of a search condition.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:CONDition {IN|OFF|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 search mode of a search condition.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:MODE {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 DLM2000 Features Guide for this information.

Example
```
:HISTORY:SEARCH:NUMBER1:PARAMETER:LIM
IT 0,1
:HISTORY:SEARCH:NUMBER1:PARAMETER:LIM
IT?
-> :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 for 2-channel models)
<y> of BIT<y> = 1 to 8
<y> of FFT<y> = 1 or 2 (1 only for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)
<y> of XY<y> = 1 or 2 (1 only for 2-channel models)

Example
```
:HISTORY:SEARCH:NUMBER1:PARAMETER:TRA
CE 1
:HISTORY:SEARCH:NUMBER1:PARAMETER:TRA
CE?
-> :HISTORY:SEARCH:NUMBER1:PARAMETER:
TRACE 1
```

:HISTory[:SEARch]:NUMBer<x>:PARameter:VALue?

Function Queries the specified waveform parameter's measured value for parameter searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:
PARameter:VALue?
```
<x> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:PARAMETER:VAL
UE?
-> :HISTORY:SEARCH:NUMBER1:PARAMETER:
VALUE 1.98E-03
```

Description When the mode is set to OFF or when the value is otherwise immeasurable, the DLM2000 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 that will be used for polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:H
POSition {<NRf>}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:H
POSition?
```
<x> = 1 to 4
<NRf> = −5 to 5 (divisions)

Example
```
:HISTORY:SEARCH:NUMBER1:POLYGON:HPOSI
TION 1
:HISTORY:SEARCH:NUMBER1:POLYGON:HPOSI
TION?
-> :HISTORY:SEARCH:NUMBER1:POLYGON:HP
OSITION 1.000E+00
```

Description When TRACe is set to XY<x>, the <NRf> is 1 to 4 (div).

:HISTory[:SEARch]:NUMBer<x>:POLYgon:RANGe

Function Sets or queries the source window that will be used for polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:R
ANGe {MAIN|Z1|Z2}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:R
ANGe?
```
<x> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:POLYGON:RAN
GE MAIN
:HISTORY:SEARCH:NUMBER1:POLYGON:RAN
GE?
-> :HISTORY:SEARCH:NUMBER1:POLYGON:RA
NGE MAIN
```

:HISTory[:SEARch]:NUMBer<x>:POLYgon:TRACe

Function Sets or queries the source waveform that will be used for 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 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

<y> of XY<y> = 1 or 2 (1 only for 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 that will be used for polygonal zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:POLYgon:VPOSition {<NRf>}
:HISTory[:SEARch]:NUMBer<x>:POLYgon:VPOSition?
```
<x> = 1 to 4

<NRf> = −4 to 4 (divisions)

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 that will be used for 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 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 that will be used for rectangular zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:
RECTangle:HORizontal {<NRf>,<NRf>}
:HISTory[:SEARch]:NUMBer<x>:
RECTangle:HORizontal?
```
<x> = 1 to 4

<NRf> = −5 to 5 (divisions)

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 set to XY<x>, the <NRf> is 1 to 4 (div).

:HISTory[:SEARch]:NUMBer<x>:RECTangle:RANGe

Function Sets or queries the source window of the rectangle that will be used for 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 that will be used for 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 for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)
<y> of XY<y> = 1 or 2 (1 only for 2-channel models)

Example
```
:HISTORY:SEARCH:NUMBER1:RECTANGLE:TRA
CE 1
:HISTORY:SEARCH:NUMBER1:RECTANGLE:TRA
CE?
-> :HISTORY:SEARCH:NUMBER1:RECTANGLE:
TRACE 1
```

:HISTory[:SEARch]:NUMBer<x>:RECTangle:VERTical

Function Sets or queries the vertical position of the rectangle that will be used for rectangular zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:
RECTangle:VERTical {<NRf>,<NRf>}
:HISTory[:SEARch]:NUMBer<x>:
RECTangle:VERTical?
```
<x> = 1 to 4
<NRf> = −4 to 4 (divisions)

Example
```
:HISTORY:SEARCH:NUMBER1:RECTANGLE:VER
TICAL 1,2
:HISTORY:SEARCH:NUMBER1:RECTANGLE:VER
TICAL?
-> :HISTORY:SEARCH:NUMBER1:RECTANGLE:
VERTICAL 2.000E+00,1.000E+00
```

:HISTory[:SEARch]:NUMBer<x>:WAVE?

Function Queries all 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 that will be used for waveform zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE:RAN
Ge {MAIN|Z1|Z2}
:HISTory[:SEARch]:NUMBer<x>:WAVE:RAN
Ge?
```
<x> = 1 to 4

Example
```
:HISTORY:SEARCH:NUMBER1:WAVE:RAN
GE MAIN
:HISTORY:SEARCH:NUMBER1:WAVE:RANGE?
-> :HISTORY:SEARCH:NUMBER1:WAVE:RAN
GE MAIN
```

:HISTory[:SEARch]:NUMBer<x>:WAVE:TRACe

Function Sets or queries the source waveform that will be used for waveform zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRA
Ce {<NRf>|MATH<y>}
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRA
Ce?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

Example
```
:HISTORY:SEARCH:NUMBER1:WAVE:TRACE 1
:HISTORY:SEARCH:NUMBER1:WAVE:TRACE?
-> :HISTORY:SEARCH:NUMBER1:WAVE:TRA
CE 1
```

:HISTory[:SEARch]:NUMBer<x>:WAVE:TRANge

Function Sets or queries the determination area that will be used for waveform zone searching.

Syntax
```
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRAN
ge {<NRf>,<NRf>}
:HISTory[:SEARch]:NUMBer<x>:WAVE:TRAN
ge?
```
<NRf> = −5 to 5 (divisions)

Example
```
:HISTORY:SEARCH:NUMBER1:WAVE:TRAN
GE 1,2
:HISTORY:SEARCH:NUMBER1:WAVE:TRANGE?
-> :HISTORY:SEARCH:NUMBER1:WAVE:TRAN
GE 2.000E+00,1.000E+00
```

:HISTory[:SEARch]:NUMBer<x>:WAVE:ZNUMber

Function Sets or queries the zone number that will be used for 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 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)
<x> of XY<x> = 1 or 2 (1 only for 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 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 image data output settings.

Syntax `:IMAGe?`

:IMAGe:ABORt

Function Aborts saving screen image data to the storage medium.

Syntax `:IMAGe:ABORt`

Example `:IMAGE:ABORT`

:IMAGe:BACKground

Function Sets or queries the screen image background.

Syntax
```
:IMAGe:BACKground {NORMal|TRANsparent
}
:IMAGe:BACKground?
```

Example
```
:IMAGE:BACKGROUND NORMAL
:IMAGE:BACKGROUND?
-> :IMAGE:BACKGROUND NORMAL
```

:IMAGe:COMMent

Function Sets or queries the comment at the lower right of the screen.

Syntax
```
:IMAGe:COMMent {<String>}
:IMAGe:COMMent?
```
<String> = Up to 32 characters

Example
```
:IMAGE:COMMENT "THIS IS TEST."
:IMAGE:COMMENT?
-> :IMAGE:COMMENT "THIS IS TEST."
```

:IMAGe:EXECute

Function Saves screen image data to a storage medium.

Syntax `:IMAGe:EXECute`

Example `:IMAGE:EXECUTE`

:IMAGe:FORMat

Function Sets or queries the screen image 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 image 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 status 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 DLM2000 Features Guide for this information.

Example `:IMAGE:SAVE:CDIRECTORY "ABC"`

:IMAGe:SAVE:DRIVe

Function Sets the medium to create files on.

Syntax
```
:IMAGe:SAVE:DRIVe {FLAShmem|NETWork|
USB,<NRf>}
```
<NRf> = 0 to 3

Example `:IMAGE:SAVE:DRIVE FLASHMEM`

Description You can omit the <NRf> for USB if the drive is not partitioned or is not divided into LUNs.

:IMAGe:SAVE:NAME

Function Sets or queries the file name for the file that will be created.

Syntax
```
:IMAGe:SAVE:NAME {<File name>}
:IMAGe:SAVE:NAME?
```

Example
```
:IMAGE:SAVE:NAME "DISP_1"
:IMAGE:SAVE:NAME?
-> :IMAGE:SAVE:NAME "DISP_1"
```

:IMAGe:SEND?

Function Queries the screen image data value.

Syntax `:IMAGe:SEND?`

Example `:IMAGE:SEND?`

`-> :IMAGE:SEND #8`(number of bytes, 8 digits)(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 image 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 initialization.

Syntax `:INITialize:UNDO`

Example `:INITIALIZE:UNDO`

5.18 LOGic Group

You cannot use commands that relate to logic group on models that are not equipped with logic inputs.

:LOGic?

Function Queries all logic input waveform settings.

Syntax
```
:LOGic?
```

:LOGic:MODE

Function Sets or queries the on/off status of logic input.

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]:ALL?

Function Queries all logic input bit settings.

Syntax
```
:LOGic[:PODA]:ALL?
```

:LOGic[:PODA]:ALL:DISPlay

Function Sets the display on/off status of all bits of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:ALL:DISPlay {<Boolean>}
:LOGic[:PODA]:ALL:DISPlay?
```

Example
```
:LOGIC:PODA:ALL:DISPLAY ON
:LOGIC:PODA:ALL:DISPLAY?
-> :LOGIC:PODA:ALL:DISPLAY 1
```

:LOGic[:PODA]:ALL:LEVel

Function Sets or queries the user-defined threshold level of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:ALL:LEVel {<Voltage>}
:LOGic[:PODA]:ALL:LEVel?
```
<Voltage> = −10 to 10 V (in 0.1-V steps)

Example
```
:LOGIC:PODA:ALL:LEVEL 1V
:LOGIC:PODA:ALL:LEVEL?
-> :LOGIC:PODA:ALL:LEVEL 1.0E+00
```

:LOGic[:PODA]:ALL:TYPE

Function Selects the threshold level of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:ALL:TYPE {CMOS1|CMOS2|
CMOS3|CMOS5|ECL}
```

Example
```
:LOGIC:PODA:ALL:TYPE CMOS1
```

:LOGic[:PODA]:BIT<x>?

Function Queries all of the settings for a bit of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:BIT<x>?
```
<x> = 1 to 8

:LOGic[:PODA]:BIT<x>:DISPlay

Function Sets the display on/off status of all bits of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:BIT<x>:DISPlay {<Boolea
n>}
:LOGic[:PODA]:BIT<x>:DISPlay?
```
<x> = 1 to 8

Example
```
:LOGIC:PODA:BIT1:DISPLAY ON
:LOGIC:PODA:BIT1:DISPLAY?
-> :LOGIC:PODA:BIT1:DISPLAY 1
```

:LOGic[:PODA]:BIT<x>:LABel

Function Sets or queries the label of a bit of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:BIT<x>:LABel {<String>}
:LOGic[:PODA]:BIT<x>:LABel?
```
<x> = 1 to 8

<String> = Up to eight characters

Example
```
:LOGIC:PODA:BIT1:LABEL "NO_1"
:LOGIC:PODA:BIT1:LABEL?
-> :LOGIC:PODA:BIT1:LABEL "NO_1"
```

:LOGic[:PODA]:BIT<x>:LEVel

Function Sets or queries the user-defined threshold level for a bit of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:BIT<x>:LEVel {<Voltage>
}
:LOGic[:PODA]:BIT<x>:LEVel?
```
<x> = 1 to 8

<Voltage> = −10 to 10 V (in 0.1-V steps)

Example
```
:LOGIC:PODA:BIT1:LEVEL 1V
:LOGIC:PODA:BIT1:LEVEL?
-> :LOGIC:PODA:BIT1:LEVEL 1.0E+00
```

:LOGic[:PODA]:BIT<x>:TYPE

Function Selects the threshold level for a bit of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:BIT<x>:TYPE {CMOS1|
CMOS2|CMOS3|CMOS5|ECL}
```
<x> = 1 to 8

Example
```
:LOGIC:PODA:BIT1:TYPE CMOS1
```

:LOGic[:PODA]:DESKew

Function Sets or queries the deskewing of a specified logic input pod.

Syntax
```
:LOGic[:PODA]:DESKew {<Time>}
:LOGic[:PODA]:DESKew?
```
<Time> = −100 to 100 ns (in 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 a specified logic input pod.

Syntax
```
:LOGic[:PODA]:HYSTeresis {HIGH|LOW}
:LOGic[:PODA]:HYSTeresis?
```

Example
```
:LOGIC:PODA:HYSTERESIS HIGH
:LOGIC:PODA:HYSTERESIS?
-> :LOGIC:PODA:HYSTERESIS HIGH
```

:LOGic:GROup?

Function Queries all logic settings.

Syntax
```
:LOGic:GROup?
```

:LOGic:GROup:BITOrder

Function Sets or queries the logic bit order.

Syntax
```
:LOGic:GROup:BITOrder {BIT7_BIT0|
BIT0_BIT7}
:LOGic:GROup:BITOrder?
```

Example
```
:LOGIC:GROUP:BITORDER BIT7_BIT0
:LOGIC:GROUP:BITORDER?
-> :LOGIC:GROUP:BITORDER BIT7_BIT0
```

:LOGic:GROup:BUNDle?

Function Queries all logic bundle settings.

Syntax
```
:LOGic:GROup:BUNDle?
```

:LOGic:GROup:BUNDle[:MODE]

Function Sets or queries the logic bundle mode.

Syntax
```
:LOGic:GROup:BUNDle[:MODE] {<Boolean>
}
:LOGic:GROup:BUNDle[:MODE]?
```

Example
```
:LOGIC:GROUP:BUNDLE:MODE ON
:LOGIC:GROUP:BUNDLE:MODE?
-> :LOGIC:GROUP:BUNDLE:MODE 1
```

:LOGic:GROup:FORMat

Function Sets or queries the logic bundle display format (bus display).

Syntax
```
:LOGic:GROup:FORMat {BINary|HEXa}
:LOGic:GROup:FORMat?
```

Example
```
:LOGIC:GROUP:FORMAT BINARY
:LOGIC:GROUP:FORMAT?
-> :LOGIC:GROUP:FORMAT BINARY
```

:LOGic:GROup:STATe?

Function Queries all logic-input state display settings.

Syntax
```
:LOGic:GROup:STATe?
```

:LOGic:GROup:STATe:CLOCk

Function Sets or queries the reference clock waveform of the logic-input state display.

Syntax
```
:LOGic:GROup:STATe:CLOCk {<NRf>|
BIT<x>}
:LOGic:GROup:STATe:CLOCk?
```
<NRf> = 1 to 3
<x> = 1 to 8

Example
```
:LOGIC:GROUP:STATE:CLOCK 1
:LOGIC:GROUP:STATE:CLOCK?
-> :LOGIC:GROUP:STATE:CLOCK 1
```

:LOGic:GROup:STATe:HYSTeresis

Function Sets or queries the hysteresis of the reference clock waveform of the logic-input state display.

Syntax
```
:LOGic:GROup:STATe:HYSTeresis {<NRf>}
:LOGic:GROup:STATe:HYSTeresis?
```
<NRf> = 0.0 to 4.0 (divisions)

Example
```
:LOGIC:GROUP:STATE:HYSTERESIS 1.0
:LOGIC:GROUP:STATE:HYSTERESIS?
-> :LOGIC:GROUP:STATE:HYSTERESIS 1.00
0E+00
```

:LOGic:GROup:STATe:MODE

Function Sets or queries the on/off status of the logic-input state display.

Syntax
```
:LOGic:GROup:STATe:MODE {<Boolean>}
:LOGic:GROup:STATe:MODE?
```

Example
```
:LOGIC:GROUP:STATE:MODE ON
:LOGIC:GROUP:STATE:MODE?
-> :LOGIC:GROUP:STATE:MODE 1
```

:LOGic:GROup:STATe:POLarity

Function Sets or queries the polarity of the reference clock waveform of the logic-input state display.

Syntax
```
:LOGic:GROup:STATe:POLarity {RISE|
FALL|BOTH}
:LOGic:GROup:STATe:POLarity?
```

Example
```
:LOGIC:GROUP:STATE:POLARITY RISE
:LOGIC:GROUP:STATE:POLARITY?
-> :LOGIC:GROUP:STATE:POLARITY RISE
```

:LOGic:GROup:STATe:THReshold

Function Sets or queries the detection level of the reference clock waveform of the logic-input state display.

Syntax
```
:LOGic:GROup:STATe:THReshold {<NRf>}
:LOGic:GROup:STATe:THReshold?
```
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:LOGIC:GROUP:STATE:THRESHOLD 1
:LOGIC:GROUP:STATE:THRESHOLD?
-> :LOGIC:GROUP:STATE:THRESHOLD 1.000
E+00
```

:LOGic:POSition

Function Sets or queries the vertical position of the logic signal.

Syntax
```
:LOGic:POSition {<NRf>}
:LOGic:POSition?
```
<NRf> = −7 to 39

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|MIDium|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 or 2 (1 only for 2-channel models)

:MATH<x>:DISPlay

Function Sets or queries whether or not computed waveforms will be displayed (on/off).

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

:MATH<x>:ECOunt:HYSTeresis

Function Sets or queries the hysteresis for the edge detection level of edge counting.

Syntax
```
:MATH<x>:ECOunt:HYSTeresis {<NRf>}
:MATH<x>:ECOunt:HYSTeresis?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 0 to 4 (divisions)

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 counting.

Syntax
```
:MATH<x>:ECOunt:POLarity {FALL|RISE}
:MATH<x>:ECOunt:POLarity?
```
<x> = 1 or 2 (1 only for 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 or 2 (1 only for 2-channel models)
<Voltage>, <Current> = See the DLM2000 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 or 2 (1 only for 2-channel models)

:MATH<x>:FILTer:FORDer

Function Sets or queries the filter order of an IIR filter.

Syntax
```
:MATH<x>:FILTer:FORDer {<NRf>}
:MATH<x>:FILTer:FORDer?
```
<x> = 1 or 2 (1 only for 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 a high-pass IIR filter.

Syntax
```
:MATH<x>:FILTer:HCUToff {<Frequency>}
:MATH<x>:FILTer:HCUToff?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Frequency> = 0.01 to 500 M(Hz)

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 a low-pass IIR filter.

Syntax
```
:MATH<x>:FILTer:LCUToff {<Frequency>}
:MATH<x>:FILTer:LCUToff?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Frequency> = 0.01 to 500 M(Hz)

Example
```
:MATH1:FILTER:LCUTOFF 10HZ
:MATH1:FILTER:LCUTOFF?
-> :MATH1:FILTER:LCUTOFF 10.00E+00
```

:MATH<x>:FILTer:TIME

Function Sets or queries a phase shift delay time.

Syntax
```
:MATH<x>:FILTer:TIME {<Time>}
:MATH<x>:FILTer:TIME?
```
<x> = 1 or 2 (1 only for 2-channel models)
<Time> = See the DLM2000 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 a filter type.

Syntax
```
:MATH<x>:FILTer:TYPE {DELay|MAVG|
IHPass|ILPass}
:MATH<x>:FILTer:TYPE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MATH1:FILTER:TYPE DELAY
:MATH1:FILTER:TYPE?
-> :MATH1:FILTER:TYPE DELAY
```

:MATH<x>:FILTer:WEIGht

Function Sets or queries a moving average weight.

Syntax
```
:MATH<x>:FILTer:WEIGht {<NRf>}
:MATH<x>:FILTer:WEIGht?
```
<x> = 1 or 2 (1 only for 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 integration settings.

Syntax
```
:MATH<x>:INTegral?
```
<x> = 1 or 2 (1 only for 2-channel models)

:MATH<x>:INTegral:SPOint

Function Sets or queries the integration start position.

Syntax
```
:MATH<x>:INTegral:SPOint {<NRf>}
:MATH<x>:INTegral:SPOint?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:MATH1:INTEGRAL:SPOINT 1.5
:MATH1:INTEGRAL:SPOINT?
-> :MATH1:INTEGRAL:SPOINT 1.5000000
```

:MATH<x>:LABel?

Function Queries all computed waveform label settings.

Syntax
```
:MATH<x>:LABel?
```
<x> = 1 or 2 (1 only for 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 or 2 (1 only for 2-channel models)
<String> = Up to eight characters

Example
```
:MATH1:LABEL:DEFINE "MATH1"
:MATH1:LABEL:DEFINE?
-> :MATH1:LABEL:DEFINE "MATH1"
```

:MATH<x>:LABel:MODE

Function Sets or queries the display on/off status of a computed waveform label.

Syntax
```
:MATH<x>:LABel:MODE {<Boolean>}
:MATH<x>:LABel:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MATH1:LABEL:MODE ON
:MATH1:LABEL:MODE?
-> :MATH1:LABEL:MODE 1
```

:MATH<x>:OPERation

Function Sets or queries an operator.

Syntax
```
:MATH<x>:OPERation {ECOunt|FILTer|
INTegral|MINus|MULTiple|PLUS|RCOunt|
USERdefine},{<NRf>|MATH1}[,{<NRf>|
MATH1}]
:MATH<x>:OPERation?
```
<x> = 1 or 2 (1 only for 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), set the source waveform in the first <NRf>.
- For binary operators (MINus|MULTiple|PLUS|RCOunt), set the source waveform of the first term in the first <NRf> and the source waveform of the second term in the second <NRf>.
- For user-defined operators, you do not need to use <NRf>.

:MATH:RCOunt?

Function Queries all rotary-count computation settings.

Syntax
```
:MATH:RCOunt?
```

:MATH<x>:RCOunt:THReshold<y>

Function Sets or queries the threshold level for rotary-count computation.

Syntax
```
:MATH<x>:RCOunt:THReshold<y> {<NRf>|
<Voltage>|<Current>}
:MATH<x>:RCOunt:THReshold<y>?
```
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2
<Voltage>, <Current> = See the DLM2000 Features Guide for this information.

Example
```
:MATH1:RCOUNT:THRESHOLD1 1
:MATH1:RCOUNT:THRESHOLD1?
-> :MATH1:RCOUNT:THRESHOLD1 1.000E+00
```

:MATH<x>:SCALe?

Function Queries all scaling settings.

Syntax
```
:MATH<x>:SCALe?
```
<x> = 1 or 2 (1 only for 2-channel models)

:MATH<x>:SCALe:CENTer

Function Sets or queries a center value for manual scaling.

Syntax
```
:MATH<x>:SCALe:CENTer {<NRf>}
:MATH<x>:SCALe:CENTer?
```
<x> = 1 or 2 (1 only for 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 a scaling mode.

Syntax
```
:MATH<x>:SCALe:MODE {AUTO|MANual}
:MATH<x>:SCALe:MODE?
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MATH1:SCALE:MODE AUTO
:MATH1:SCALE:MODE?
-> :MATH1:SCALE:MODE AUTO
```

:MATH<x>:SCALe:SENSitivity

Function Sets or queries a span from the center value for manual scaling.

Syntax
```
:MATH<x>:SCALe:SENSitivity {<NRf>}
:MATH<x>:SCALe:SENSitivity?
```
<x> = 1 or 2 (1 only for 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 or 2 (1 only for 2-channel models)

:MATH<x>:UNIT[:DEFine]

Function Sets or queries a unit of computation.

Syntax
```
:MATH<x>:UNIT[:DEFine] {<String>}
:MATH<x>:UNIT[:DEFine]?
```
<x> = 1 or 2 (1 only for 2-channel models)
<String> = Up to four characters

Example
```
:MATH1:UNIT:DEFINE "EU"
:MATH1:UNIT:DEFINE?
-> :MATH1:UNIT:DEFINE "EU"
```

Description Units are applied to scale values. They never affect the computation result.

:MATH<x>:UNIT:MODE

Function Sets or queries whether a unit of computation will be attached automatically or manually.

Syntax
```
:MATH<x>:UNIT:MODE {AUTO|USERdefine}
:MATH<x>:UNIT:MODE?
```
<x> = 1 or 2 (1 only for 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 or 2 (1 only for 2-channel models)

:MATH<x>:USERdefine:AVERage?

Function Queries all averaging settings for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:AVERage?
```
<x> = 1 or 2 (1 only for 2-channel models)

Description The command affects both MATH1 and MATH2.

:MATH<x>:USERdefine:AVERage:EWEight

Function Sets or queries the attenuation constant of exponential averaging in user-defined computation.

Syntax `:MATH<x>:USERdefine:AVERage:EWEight {<NRf>}`
`:MATH<x>:USERdefine:AVERage:EWEight?`
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 2 to 1024 in 2^n steps)

Example `:MATH1:USERDEFINE:AVERAGE:EWEIGHT 2`
`:MATH1:USERDEFINE:AVERAGE:EWEIGHT?`
`-> :MATH1:USERDEFINE:AVERAGE:EWEIGHT 2`

Description The command affects both MATH1 and MATH2.

:MATH<x>:USERdefine:AVERage:MODE

Function Sets or queries the averaging mode for user-defined computation.

Syntax `:MATH<x>:USERdefine:AVERage:MODE {EXPonent|OFF}`
`:MATH<x>:USERdefine:AVERage:MODE?`
<x> = 1 or 2 (1 only for 2-channel models)

Example `:MATH1:USERDEFINE:AVERAGE:MODE EXPONENT`
`:MATH1:USERDEFINE:AVERAGE:MODE?`
`-> :MATH1:USERDEFINE:AVERAGE:MODE EXPONENT`

Description The command affects both MATH1 and MATH2.

: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 or 2 (1 only for 2-channel models)
<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 The command affects both MATH1 and MATH2.

:MATH<x>:USERdefine:DEFine

Function Sets or queries the equation for user-defined computation.

Syntax `:MATH<x>:USERdefine:DEFine {<String>}`
`:MATH<x>:USERdefine:DEFine?`
<x> = 1 or 2 (1 only for 2-channel models)
<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 that appear on the DLM2000 soft keyboard.

:MATH<x>:USERdefine:FILTer<y>?

Function Queries all filter settings for user-defined computation.

Syntax `:MATH<x>:USERdefine:FILTer<y>?`
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Description The command affects both MATH1 and MATH2.

: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>:BAND {BPASs|HPASs|LPASs}`
`:MATH<x>:USERdefine:FILTer<y>:BAND?`
<x> = 1 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example `:MATH1:USERDEFINE:FILTER1:BAND BPASS`
`:MATH1:USERDEFINE:FILTER1:BAND?`
`-> :MATH1:USERDEFINE:FILTER1:BAND BPASS`

Description The command affects both MATH1 and MATH2.

:MATH<x>:USERdefine:FILTer<y>:CUToff<z>

Function Sets or queries the cutoff frequency for user-defined computation.

Syntax `:MATH<x>:USERdefine:FILTer<y>:CUToff<z> {<NRf>}`
`:MATH<x>:USERdefine:FILTer<y>:CUToff<z>?`
<x> = 1 or 2 (1 only for 2-channel models)
<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:CUTOFF1 10.0`

Description CUToff2 can only be specified for ":MATH<x>:USERdefine:FILTer<x>:BAND BPASs."
The command affects both MATH1 and MATH2.

: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 or 2 (1 only for 2-channel models)
<y> = 1 or 2

Example
```
:MATH1:USERDEFINE:FILTER1:TYPE IIR
:MATH1:USERDEFINE:FILTER1:TYPE?
-> :MATH1:USERDEFINE:FILTER1:TYPE IIR
```

Description The command affects both MATH1 and MATH2.

:MATH<x>:USERdefine:HISTory:ABORt

Function Aborts user-defined computation (Math on History) of the history waveform.

Syntax
```
:MATH<x>:USERdefine:HISTory:ABORt
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MATH1:USERDEFINE:HISTORY:ABORT
```

:MATH<x>:USERdefine:HISTory:EXECute

Function Executes user-defined computation (Math on History) of the history waveform.

Syntax
```
:MATH<x>:USERdefine:HISTory:EXECute
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MATH1:USERDEFINE:HISTORY:EXECUTE
```

:MATH<x>:USERdefine:SCALe?

Function Queries all scaling settings for user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe?
```
<x> = 1 or 2 (1 only for 2-channel models)

:MATH<x>:USERdefine:SCALe:ARANging

Function Executes auto ranging of user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe:ARANging
```
<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MATH1:USERDEFINE:SCALE:ARANGING
```

:MATH<x>:USERdefine:SCALe:CENTer

Function Sets or queries the center value for scaling in user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe:CENT
er {<NRf>}
:MATH<x>:USERdefine:SCALe:CENTer?
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = −1.0000E+31 to 1.0000E+31

Example
```
:MATH1:USERDEFINE:SCALE:CENTER 1
:MATH1:USERDEFINE:SCALE:CENTER?
-> :MATH1:USERDEFINE:SCALE:CENTER 1.0
0000E+00
```

:MATH<x>:USERdefine:SCALe:SENSitivity

Function Sets or queries the span from the center value for scaling in user-defined computation.

Syntax
```
:MATH<x>:USERdefine:SCALe:SENSitivi
ty {<NRf>}
:MATH<x>:USERdefine:SCALe:SENSitivi
ty?
```
<x> = 1 or 2 (1 only for 2-channel models)
<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

You cannot use commands that relate to logic waveforms on models that are not equipped with logic inputs.

:MEASure?

Function Queries all of the settings for automated measurement of waveform parameters.

Syntax `:MEASure?`

:MEASure:BIT<x>?

Function Queries the on/off status of all parameters of a 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 of the on/off status of a parameter 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 the setting of a waveform parameter 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 Below is an example for 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 Below is an example for 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 DLM2000 returns “NAN” (not a number).

:MEASure:BIT<x>:AREA2:<Parameter>:STATe

Function Sets or queries the on/off status 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 Below is an example for 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 an automatically measured value of a logic waveform parameter in Area2.

Syntax `:MEASure:BIT<x>:AREA2:<Parameter>:VALue? [{NRf}]`
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}
<NRf> = See the DLM2000 Features Guide for this information.

Example Below is an example for 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 DLM2000 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 DLM2000 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 status of all measurement items of a logic waveform in Area2 to all other waveforms.

Syntax `:MEASure:BIT<x>:AREA2:COPY`
<x> = 1 to 8

Example `:MEASURE:BIT1:AREA2:COPY`

:MEASure:BIT<x>:AREA2:DELay?

Function Queries all of the settings for measuring 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 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 reference waveform settings used to measure the delay between the specified logic waveform and the reference 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 edge count of the reference waveform used to measure the delay between the specified logic waveform and the reference 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:C
OUNT 2
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:C
OUNT?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENC
E:COUNT 2
```

:MEASure:BIT<x>:AREA2:DELay:REFerence:SLOPe

Function Sets or queries the slope of the reference waveform used to measure the delay between the specified logic waveform and the reference 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:S
LOPE FALL
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:S
LOPE?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENC
E: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 of Area2 will be set to the trigger point or a waveform.

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:S
OURCE TRACE
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:S
OURCE?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENC
E:SOURCE TRACE
```

:MEASure:BIT<x>:AREA2:DELay:REFerence:TRACe

Function Sets or queries the reference waveform edge used to measure the delay between the specified logic waveform and the reference 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:T
RACE 2
:MEASURE:BIT1:AREA2:DELAY:REFERENCE:T
RACE?
-> :MEASURE:BIT1:AREA2:DELAY:REFERENC
E:TRACE 2
```

:MEASure:BIT<x>:AREA2:DELay:STATe

Function Sets or queries the on/off status and the display format of a delay parameter in Area2.

Syntax
```
:MEASure:BIT<x>:AREA2:DELay:STATe {OF
F|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 the setting of a waveform parameter 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 Below is an example for 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 Below is an example for the average frequency of bit 1.

```
:MEASURE:BIT1:AVGFREQ:MAXIMUM?
-> :MEASURE:BIT1:AVGFREQ:MAXIMUM 10.0
000E+03
```

Description If the statistical value is immeasurable, the DLM2000 returns “NAN” (not a number).

:MEASure:BIT<x>:<Parameter>:STATe

Function Sets or queries the on/off status 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 Below is an example for 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 an automatically measured value of a logic waveform parameter.

Syntax :MEASure:BIT<x>:<Parameter>:VALue? [{NRf}]
<x> = 1 to 8
<Parameter> = {AVGFreq|DELay|DUTYcycle|FREQuency|PERiod|PNUMber}
<NRf> = See the DLM2000 Features Guide for this information.

Example Below is an example for the average frequency of bit 1.

```
:MEASURE:BIT1:AVGFREQ:VALUE?
-> :MEASURE:BIT1:AVGFREQ:VALUE 10.000
0E+03
```

Description If the value is immeasurable, the DLM2000 returns “NAN” (not a number). The “<NRf>” at the end is used to specify what parameter value to query. It indicates the order of the parameter value since statistical processing began. If the specified parameter does not exist, the DLM2000 returns “NAN” (not a number).

- You do not have to enter a value for <NRf> for statistics other than the period. If you do not enter a value, the most recent history parameter value will be queried. If you enter a value for <NRf>, the order starts with the newest waveform in the memory and increases as the waveform parameter values get older.
- You do not have to enter a value for <NRf> when cyclic statistical processing is finished. If you do not enter a value, the parameter value within the period that was measured last will be queried. If you enter a value for <NRf>, the parameter value within the period that was measured <NRf> times after the measurement shown on the screen left edge will be queried.

:MEASure:BIT<x>:COPY

Function Copies the on/off status 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 of the settings for measuring 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 of 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 count number of a source waveform for measuring the delay between channels of a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:MEASure:COUNt {<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 the slope of a source waveform for measuring the delay between channels of a logic waveform.

Syntax
```
:MEASure:BIT<x>:DELay:MEASure:SLOPe {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 reference waveform settings used to measure the delay between the specified logic waveform and the reference waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence?
```
<x> = 1 to 8

:MEASure:BIT<x>:DELay:REFerence:COUNt

Function Sets or queries the edge count of the reference waveform used to measure the delay between the specified logic waveform and the reference waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:COUNt {<NRf>}
:MEASure:BIT<x>:DELay:REFerence:COUNt?
```
<x> = 1 to 8
<NRf> = 1 to 10

Example
```
:MEASURE:BIT1:DELAY:REFERENCE:COUNT 2
:MEASURE:BIT1:DELAY:REFERENCE:COUNT?
-> :MEASURE:BIT1:DELAY:REFERENCE:COUNT 2
```

:MEASure:BIT<x>:DELay:REFerence:SLOPe

Function Sets or queries the slope of the reference waveform used to measure the delay between the specified logic waveform and the reference waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:SLOPe {FALL|RISE}
:MEASure:BIT<x>:DELay:REFerence:SLOPe?
```
<x> = 1 to 8

Example
```
:MEASURE:BIT1:DELAY:REFERENCE:SLOPE FALL
:MEASURE:BIT1:DELAY:REFERENCE:SLOPE?
-> :MEASURE:BIT1:DELAY:REFERENCE:SLOPE 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 waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:SOURce {TRACe|TRIGger}
:MEASure:BIT<x>:DELay:REFerence:SOURce?
<x> = 1 to 8 Example :MEASURE:BIT1:DELAY:REFERENCE:SOURCE TRACE
:MEASURE:BIT1:DELAY:REFERENCE:SOURCE?
-> :MEASURE:BIT1:DELAY:REFERENCE:SOURCE TRACE
```

:MEASure:BIT<x>:DELay:REFerence:TRACe

Function Sets or queries the reference waveform edge used to measure the delay between the specified logic waveform and the reference waveform.

Syntax
```
:MEASure:BIT<x>:DELay:REFerence:TRACe {<NRf>|BIT<y>|MATH<y>}
:MEASure:BIT<x>:DELay:REFerence:TRACe?
```
<x> = 1 to 8
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:BIT1:DELAY:REFERENCE:TRACE 2
:MEASURE:BIT1:DELAY:REFERENCE:TRACE?
-> :MEASURE:BIT1:DELAY:REFERENCE:TRACE 2
```

:MEASure:BIT<x>:DELay:STATe

Function Sets or queries the on/off status and the display format of a 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 the on/off status of all parameter of a waveform.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example `:MEASURE:CHANNEL1:ALL ON`

:MEASure:{CHANnel<x>|MATH<x>}:AREA2?

Function Queries the on/off status of all parameter of a waveform in Area2.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:AREA2?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example `:MEASURE:CHANNEL1:AREA2:ALL ON`

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>?

Function Queries the setting of a waveform parameter of a waveform in Area2.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>?`
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 Below is an example for the average 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 Below is an example for the average of CH1.

```
:MEASURE:CHANNEL1:AREA2:AVERAGE:MAXIMUM?
-> :MEASURE:CHANNEL1:AREA2:AVERAGE:MAXIMUM 10.0000E+03
```

Description If the statistical value is immeasurable, the DLM2000 returns “NAN” (not a number).

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:<Parameter>:STATe

Function Sets or queries the on/off status 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 Below is an example for the average 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 an automatically measured 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 DLM2000 Features Guide for this information.

Example Below is an example for the average of CH1.

```
:MEASURE:CHANNEL1:AREA2:AVERAGE:VALUE?
-> :MEASURE:CHANNEL1:AREA2:AVERAGE:VALUE 10.0000E+03
```

Description

- If the value is immeasurable, the DLM2000 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 DLM2000 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 last parameter “STATus,” which can be omitted, is valid when the parameter is “FALL,” “NWIDth,” “PERiod,” “PWIDth,” or “RISE.” When “STATus” is omitted, the DLM2000 returns negative values of these parameters when the measured values are less than the measurement resolution. When “,STATus” is included, the DLM2000 returns “LOW_RESOL.”

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:COPY

Function Copies the on/off status of all measurement items of a waveform in Area2 to all other waveforms.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:COPY
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:COPY
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:CYCLe

Function Sets or queries a cycle mode in 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:AREA2:CYCLE ONE
:MEASURE:CHANNEL1:AREA2:CYCLE?
-> :MEASURE:CHANNEL1:AREA2:CYCLE ONE
```

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay?

Function Queries all of the settings for measuring 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:MEASure?

Function Queries all of the settings for a source waveform 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 reference waveform settings used to measure the delay between the specified waveform and the reference waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:AREA2:DELay:REFerence:COUNt

Function Sets or queries the edge count of the reference waveform used to measure the delay between the specified waveform and the reference 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 used to measure the delay between the specified waveform and the reference 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 in Area2 will be set to the trigger point or a waveform.

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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 reference waveform edge used to measure the delay between the specified waveform and the reference 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 for 2-channel models)
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 status and the display format of a 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 the setting of a waveform parameter of a waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 Below is an example for the average 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 Below is an example for the average of CH1.
```
:MEASURE:CHANNEL1:AVERAGE:MAXIMUM?
-> :MEASURE:CHANNEL1:AVERAGE:
MAXIMUM 10.0000E+03
```

Description If the statistical value is immeasurable, the DLM2000 returns “NAN” (not a number).

:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:STATe

Function Sets or queries the on/off status 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 Below is an example for the average 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 an automatically measured value of a waveform parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:VALue? [<NRf>[,STATus]]
:MEASure:{CHANnel<x>|MATH<x>}:<Parameter>:VALue?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 DLM2000 Features Guide for this information.

Example Below is an example for the average of CH1.
```
:MEASURE:CHANNEL1:AVERAGE:VALUE?
-> :MEASURE:CHANNEL1:AVERAGE:VALUE 10.0000E+03
```

Description
- If the value is immeasurable, the DLM2000 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 DLM2000 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 last parameter “STATus,” which can be omitted, is valid when the parameter is “FALL,” “NWIDth,” “PERiod,” “PWIDth,” or “RISE.” When “STATus” is omitted, the DLM2000 returns negative values of these parameters when the measured values are less than the measurement resolution. When “,STATus” is included, the DLM2000 returns “LOW_RESOL.”

:MEASure:{CHANnel<x>|MATH<x>}:COPY

Function Copies the on/off status 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:COPY
```

:MEASure:{CHANnel<x>|MATH<x>}:CYCLe

Function Sets or queries a cycle mode.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:
CYCLe {ONE|N|OFF}
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:CYCLE ONE
:MEASURE:CHANNEL1:CYCLE?
-> :MEASURE:CHANNEL1:CYCLE ONE
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay?

Function Queries all of the settings for measuring the delay between channels of a waveform.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:DELay?`

<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure?

Function Queries all of 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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.

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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 the specified waveform and the reference waveform.

Syntax `:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence?`

<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:COUNt

Function Sets or queries the edge count of the reference waveform used to measure the delay between the specified waveform and the reference 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 for 2-channel models)

<x> of MATH<x> = 1 or 2 (1 only for 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 slope of the reference waveform used to measure the delay between the specified waveform and the reference waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:R
EFerence:SLOPe {FALL|RISE}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:R
EFerence:SLOPe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:SLO
PE FALL
:MEASURE:CHANNEL1:DELAY:REFERENCE:SLO
PE?
-> :MEASURE:CHANNEL1:DELAY:REFERENCE:
SLOPE FALL
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SOURce

Function Sets or queries whether the reference point for measuring the delay between channels of a waveform in Area2 will be set to the trigger point or a waveform in Area2.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:R
EFerence:SOURce {TRACe|TRIGger}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:R
EFerence:SOURce?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:SOU
RCE TRACE
:MEASURE:CHANNEL1:DELAY:REFERENCE:SOU
RCE?
-> :MEASURE:CHANNEL1:DELAY:REFERENCE:
SOURCE TRACE
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:TRACe

Function Sets or queries the reference waveform edge used to measure the delay between the specified waveform and the reference waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:R
EFerence:TRACe {<NRf>|MATH<x>}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:R
EFerence:TRACe?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:TRA
CE 2
:MEASURE:CHANNEL1:DELAY:REFERENCE:TRA
CE?
-> :MEASURE:CHANNEL1:DELAY:REFERENCE:
TRACE 2
```

:MEASure:{CHANnel<x>|MATH<x>}:DELay:STATe

Function Sets or queries the on/off status and the display format of a delay parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:S
TATe {OFF|ON|DEGRee}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:S
TATe?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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>}:DProxim
al?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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>}:DProxim
al:MODE {PERCent|UNIT}
:MEASure:{CHANnel<x>|MATH<x>}:DProxim
al:MODE?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:MEASURE:CHANNEL1:DPROXIMAL:MO
DE PERCENT
:MEASURE:CHANNEL1:DPROXIMAL:MODE?
-> :MEASURE:CHANNEL1:DPROXIMAL:MO
DE 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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 0 to 100% in 1% steps

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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)
<Voltage>, <NRf> = See the DLM2000 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 the probe type is set to current, this command sets or queries the <Current> value.

:MEASure:{CHANnel<x>|MATH<x>}:METHod

Function Sets or queries the high/low point setting method.

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 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 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 the automatically measured waveform parameters.

Syntax
```
:MEASure:CONTinuous:RESTart
```

Example
```
:MEASURE:CONTINUOUS:RESTART
```

:MEASure:CONTinuous:TLCHange (Trigger Level 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?
```
<x> = 1 to 4

Example
```
:MEASURE:CONTINUOUS:TLCHANGE RESTART
:MEASURE:CONTINUOUS:TLCHANGE?
-> :MEASURE:CONTINUOUS:TLCHANGE RESTART
```

:MEASure:CYCLe?

Function Queries all cyclic statistical processing settings for automated measurement.

Syntax
```
:MEASure:CYCLe?
```

:MEASure:CYCLe:ABORt

Function Aborts cyclic statistical processing for automated measurement.

Syntax
```
:MEASure:CYCLe:ABORt
```

Example
```
:MEASURE:CYCLE:ABORT
```

:MEASure:CYCLe:EXECute

Function Executes cyclic statistical processing for automated measurement.

Syntax
```
:MEASure:CYCLe:EXECute
```

Example
```
:MEASURE:CYCLE:EXECUTE
```

:MEASure:CYCLe:TRACe

Function Sets or queries the source waveform used in the cyclic statistical processing for automated measurement.

Syntax
```
:MEASure:CYCLe:TRACe {OWN|<NRf>|MATH<x>}
:MEASure:CYCLe:TRACe?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

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

:MEASure:HISTory:ABORt

Function Aborts statistical processing of automatically measured history waveforms.

Syntax `:MEASure:HISTory:ABORt`

Example `:MEASURE:HISTORY:ABORT`

:MEASure:HISTory:EXECute

Function Executes statistical processing of automatically measured history waveforms.

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 for 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)
<Parameter> = {`AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth|DELay|DUTYcycle|FALL|FREQuency|HIGH|LOW|MAXimum|MINimum|NOVershoot|NWIDth|P|PABS|PERiod|PN|POVershoot|PP|PTOPeak|PWIDth|RISE|RMS|V1|V2|WH|WHABs|WHN|WHP|Z`}

Example
```
:MEASURE:INDICATOR 1,AVERAGE
:MEASURE:INDICATOR?
-> :MEASURE:INDICATOR 1,AVERAGE
```

:MEASure:MODE

Function Sets or queries the on/off/statistical processing status of automated measurement.

Syntax
```
:MEASure:MODE {OFF|ON|CONTinuous|CYCL
e|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` sets or queries the measurement source window for normal measurement or the measurement source window of Area1 for two-area measurement.
`:Measure:RANGe2` sets or queries the measurement source window of Area2 for two-area measurement.

:MEASure:TRANge<x> (Time Range)

Function Sets or queries a time range.

Syntax
```
:MEASure:TRANge<x> {<NRf>,<NRf>}
:MEASure:TRANge<x>?
```
<x> = 1 or 2
<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:MEASURE:TRANGE1 -4,4
:MEASURE:TRANGE1?
-> :MEASURE:TRANGE1 -4.00E+00,4.00E+0
0
```

Description `:Measure:TRANge1` sets or queries the time range for normal measurement or the time range of Area1 for two-area measurement.
`:Measure:TRANge2` sets or queries the time 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 Below are examples for the maximum value.
```
:MEASURE:USER1:MAXIMUM?
-> :MEASURE:USER1:MAXIMUM 1.000E+00
```

Description If the statistical value is immeasurable, the DLM2000 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 The 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 DLM2000 Features Guide for this information.

Example
```
:MEASURE:USER1:VALUE?
-> :MEASURE:USER1:VALUE 1.000E+00
```

Description
- If the value is immeasurable, the DLM2000 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 DLM2000 returns “NAN” (not a number).
- If <NRf> is omitted, the most recent measured value is queried.

:MEASure:WAIT?

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

Syntax `:MEASure:WAIT? {<NRf>}`

<NRf> = 1 to 36000 (the timeout specified in 100 ms intervals)

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

Description
- The command returns zero if the automated measurement finishes within the specified timeout. If automated measurement does not finish, or if it was never taking place to begin with, the command returns 1.
- Even if you set a long timeout, the command will return zero as soon as automated measurement finishes.

5.21 RECall Group

:RECall:SETup<x>:EXECute

Function Recalls setup data from an internal memory area.

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 or 2 (1 only for 2-channel models)

:REFerence<x>:DISPlay

Function Sets or queries the display on/off status of a reference waveform.

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

Example
```
:REFERENCE1:DISPLAY ON
:REFERENCE1:DISPLAY?
-> :REFERENCE1:DISPLAY 1
```

:REFerence<x>:LABel?

Function Queries all label settings of a reference.

Syntax `:REFerence<x>:LABel`

<x> = 1 or 2 (1 only for 2-channel models)

:REFerence<x>:LABel[:DEFine]

Function Sets or queries a reference waveform label.

Syntax
```
:REFerence<x>:LABel[:DEFine] {<String
>}
:REFerence<x>:LABel[:DEFine]?
```
<x> = 1 or 2 (1 only for 2-channel models)

<String> = Up to eight 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 status of a reference.

Syntax
```
:REFerence<x>:LABel:MODE {<Boolean>}
:REFerence<x>:LABel:MODE?
```
<x> = 1 or 2 (1 only for 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>|MATH1}`

<x> = 1 or 2 (1 only for 2-channel models)

<NRf> = 1 to 4

Example `:REFERENCE1:LOAD 1`

:REFerence<x>:POSition

Function Sets or queries the vertical position of a reference waveform.

Syntax
```
:REFerence<x>:POSition {<NRf>}
:REFerence<x>:POSition?
```
<x> = 1 or 2 (1 only for 2-channel models)

<NRf> = −4 to 4 (divisions)

Example
```
:REFERENCE1:POSITION 1
:REFERENCE1:POSITION?
-> :REFERENCE1:POSITION 1.000E+00
```

5.23 SEARch Group

The following table indicates how the communication commands for searching correspond to the menu items that appear on the screen.

Search Type	Communication Command	Setup Menu	
		Search Menu	Search Type
Edge search	EDGE	SEARCH	Edge
Edge qualified search	QUALify		Edge Qualified
State search	PPATTern		State
Pulse width search	WIDTh		Pulse Width
State width search	PPATTern		State Width

:SEARch?

Function Queries all waveform search settings.
Syntax `:SEARch?`

:SEARch:ABORt

Function Aborts searching.
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 a 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 a 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}`
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 4 divisions 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 DLM2000 Features Guide for this information.
Example `:SEARCH:EDGE:LEVEL -10mV`
`:SEARCH:EDGE:LEVEL?`
`-> :SEARCH:EDGE:LEVEL -10.00E-03`
Description When the probe type is set to current, this command sets or queries 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 edge search source waveform.

Syntax
```
:SEARch:EDGE:SOURce {<NRf>|BIT<x>|
MATH<x>}
:SEARch:EDGE:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:SEARCH:EDGE:SOURCE 1
:SEARCH:EDGE:SOURCE?
-> :SEARCH:EDGE:SOURCE 1
```

:SEARch:EPOint

Function Sets or queries search end point.

Syntax
```
:SEARch:EPOint {<NRf>}
:SEARch:EPOint?
```
<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:SEARCH:EPOINT 0
:SEARCH:EPOINT?
-> :SEARCH:EPOINT 0.000E+00
```

:SEARch:EXECute

Function Executes searching. This is an overlap command.

Syntax
```
:SEARch:EXECute
```

Example
```
:SEARCH:EXECUTE
```

:SEARch:MARK

Function Sets or queries the on/off status of detection point marks.

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 a zoom window.

Syntax
```
:SEARch:MAG<x> {<NRf>}
:SEARch:MAG<x>?
```
<x> = 1 or 2
<NRf> = See the DLM2000 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 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:SEARCH:POSITION1 1
:SEARCH:POSITION1?
-> :SEARCH:POSITION1 1.000E+00
```

:SEARch:PPATtern?

Function Queries all state or state width search settings.

Syntax
```
:SEARch:PPATtern?
```

:SEARch:PPATtern:BITS?

Function Queries all bit settings for state or state edge searching.

Syntax
```
:SEARch:PPATtern:BITS?
```

:SEARch:PPATtern:BITS:HEXa

Function Sets the bit search requirement in hexadecimal notation for state or state width searching.

Syntax
```
:SEARch:PPATtern:BITS:HEXa {<String>}
```

Example
```
:SEARCH:PPATTERN:BITS:HEXA "AB"
```

:SEARch:PPATtern:BITS:PATTern

Function Sets or queries the bit search requirement in binary notation for state or state width searching.

Syntax
```
:SEARch:PPATtern:BITS:PATTern {<Strin
g>}
:SEARch:PPATtern:BITS:PATTern?
```

Example
```
:SEARCH:PPATTERN:BITS:PATTERN "1100xx
01"
:SEARCH:PPATTERN:BITS:PATTERN?
-> :SEARCH:PPATTERN:BITS:PATTERN "110
0xx01"
```

:SEARch:PPATtern:{CHANnel<x>|MATH<x>}?

Function Queries all of the settings of a waveform for state or state edge searching.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|
MATH<x>}?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:HYSTeresis

Function Sets or queries a source waveform hysteresis for state or state width searching.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|
MATH<x>}:HYSTeresis {<NRf>}
:SEARch:PPATtern:{CHANnel<x>|
MATH<x>}:HYSTeresis?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 0.3 to 4 divisions in 0.1 steps

Example
```
:SEARCH:PPATTERN:CHANNEL1:HYSTERES
IS 0.3
:SEARCH:PPATTERN:CHANNEL1:HYSTERESIS?
-> :SEARCH:PPATTERN:CHANNEL1:HYSTERES
IS 0.30
```

:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:LEVel

Function Sets or queries a source waveform search level for state or state width searching.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|
MATH<x>}:LEVel {<NRf>|<Voltage>}
:SEARch:PPATtern:{CHANnel<x>|
MATH<x>}:LEVel?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SEARCH:PPATTERN:CHANNEL1:LEVEL 1V
:SEARCH:PPATTERN:CHANNEL1:LEVEL?
-> :SEARCH:PPATTERN:CHANNEL1:LEVEL 1.
000E+00
```

Description When the probe type is set to current, this command sets or queries the <Current> value.

:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:PATTern

Function Sets or queries a source waveform search pattern for state or state width searching.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|MATH<x>
}:PATTern {DONTcare|HIGH|LOW}
:SEARch:PPATtern:{CHANnel<x>|MATH<x>
}:PATTern?
```
<x> of CHANnel<x> = 1 to 4 (1 or 2 for 2-channel models)
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:SEARCH:PPATTERN:CHANNEL1:PATTERN HIG
H
:SEARCH:PPATTERN:CHANNEL1:PATTERN?
-> :SEARCH:PPATTERN:CHANNEL1:
PATTERN HIGH
```

:SEARch:PPATtern:CLOCk?

Function Queries all clock channel settings for state or state width searching.

Syntax
```
:SEARch:PPATtern:CLOCk?
```

:SEARch:PPATtern:CLOCk:HYSTeresis

Function Sets or queries the clock channel hysteresis for state or state width searching.

Syntax
```
:SEARch:PPATtern:CLOCk:HYSTeresis {<N
Rf>}
:SEARch:PPATtern:CLOCk:HYSTeresis?
```
<NRf> = 0.3 to 4 divisions in 0.1 steps

Example
```
:SEARCH:PPATTERN:CLOCK:HYSTERESIS 4.0
:SEARCH:PPATTERN:CLOCK:HYSTERESIS?
-> :SEARCH:PPATTERN:CLOCK:
HYSTERESIS 4.00
```

Description If `:SEARch:PPATtern:CLOCk:SOURce` is set to NONE, an error occurs.

:SEARch:PPATtern:CLOCk:LEVel

Function Sets or queries the clock channel search level for state or state width searching.

Syntax
```
:SEARch:PPATtern:CLOCk:LEVel {<NRf>|<
Voltage>}
:SEARch:PPATtern:CLOCk:LEVel?
```
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SEARCH:PPATTERN:CLOCK:LEVEL
:SEARCH:PPATTERN:CLOCK:LEVEL?
-> :SEARCH:PPATTERN:CLOCK:LEVEL
```

Description
- If `:SEARch:PPATtern:CLOCk:SOURce` is set to NONE, an error occurs.
- When the probe type is set to current, this command sets or queries the <Current> value.

:SEARch:PPATtern:CLOCk:SLOPe

Function Sets or queries the clock channel slope for state or state width searching.

Syntax
```
:SEARch:PPATtern:CLOCk:SLOPe {FALL|RISE}
:SEARch:PPATtern:CLOCk:SLOPe?
```

Example
```
:SEARCH:PPATTERN:CLOCK:SLOPE RISE
:SEARCH:PPATTERN:CLOCK:SLOPE?
-> :SEARCH:PPATTERN:CLOCK:SLOPE RISE
```

Description If `:SEARch:PPATtern:CLOCk:SOURce` is set to NONE, an error occurs.

:SEARch:PPATtern:CLOCk:SOURce

Function Sets or queries the clock channel source waveform state or state width searching.

Syntax
```
:SEARch:PPATtern:CLOCk:SOURce {<NRf>|BIT<x>|NONE|MATH<x>}
:SEARch:PPATtern:CLOCk:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:SEARCH:PPATTERN:CLOCK:SOURCE 1
:SEARCH:PPATTERN:CLOCK:SOURCE?
-> :SEARCH:PPATTERN:CLOCK:SOURCE 1
```

:SEARch:PPATtern:LOGic

Function Sets or queries the state or state width search logic.

Syntax
```
:SEARch:PPATtern:LOGic {AND|OR}
:SEARch:PPATtern:LOGic?
```

Example
```
:SEARCH:PPATTERN:LOGIC AND
:SEARCH:PPATTERN:LOGIC?
-> :SEARCH:PPATTERN:LOGIC AND
```

:SEARch:PPATtern:POLarity

Function Sets or queries the state or state width search polarity.

Syntax
```
:SEARch:PPATtern:POLarity {ENTer|EXIT|FALSe|TRUE}
:SEARch:PPATtern:POLarity?
```

Example
```
:SEARCH:PPATTERN:POLARITY ENTER
:SEARCH:PPATTERN:POLARITY?
-> :SEARCH:PPATTERN:POLARITY ENTER
```

Description
- If you specify `:SEARch:TYPE STATe`, the valid parameter values are `{ENTer|EXIT}`.
- If you specify `:SEARch:TYPE SWIDth`, the valid parameter values are `{FALSe|TRUE}`.

:SEARch:PPATtern:TIME<x>

Function Sets or queries the pulse width setting for state or state width searching.

Syntax
```
:SEARch:PPATtern:TIME<x> {<Time>}
:SEARch:PPATtern:TIME<x>?
```
<x> = 1 or 2
<Time> = 1 ns to 500 s

Example
```
:SEARCH:PPATTERN:TIME1 1S
:SEARCH:PPATTERN:TIME1?
-> :SEARCH:PPATTERN:TIME1 1.000E+00
```

:SEARch:PPATtern:TYPE

Function Sets or queries the search type setting for state or state width searching.

Syntax
```
:SEARch:PPATtern:TYPE {BETWeen|IN|NOTBetween|OUT|TIMeout}
:SEARch:PPATtern:TYPE?
```

Example
```
:SEARCH:PPATTERN:TYPE BETWEEN
:SEARCH:PPATTERN:TYPE?
-> :SEARCH:PPATTERN:TYPE BETWEEN
```

:SEARch:QUALify?

Function Queries all edge qualified search settings.

Syntax
```
:SEARch:QUALify?
```

:SEARch:QUALify:CONDition

Function Sets or queries the edge qualified search condition.

Syntax
```
:SEARch:QUALify:CONDition {FALSe|TRUE}
:SEARch:QUALify:CONDition?
```

Example
```
:SEARCH:QUALIFY:CONDITION FALSE
:SEARCH:QUALIFY:CONDITION?
-> :SEARCH:QUALIFY:CONDITION FALSE
```

:SEARch:SELect?

Function Sets which detected point to display in the zoom window and queries the zoom position of the detected 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 DLM2000 returns “NAN.”

:SEARch:SELect? MAXimum

Function Queries the number of detected points.

Syntax `:SEARch:SELect? {MAXimum}`

Example
```
:SEARCH:SELECT? MAXIMUM
-> :SEARCH:SELECT 100
```

Description If there are no detected points, the DLM2000 returns “NAN.”

:SEARch:SKIP?

Function Queries all skip mode settings.

Syntax `:SEARch:SKIP?`

:SEARch:SKIP:DECimation

Function Sets or queries the number of detected points that will be skipped in 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 how long searching will be skipped.

Syntax
```
:SEARch:SKIP:HOLDoff {<Time>}
:SEARch:SKIP:HOLDoff?
```
<Time> = 0.1 ns to 1 s

Example
```
:SEARCH:SKIP:HOLDOFF 0S
:SEARCH:SKIP:HOLDOFF?
-> :SEARCH:SKIP:HOLDOFF 0.000E+00
```

:SEARch:SKIP:MODE

Function Sets or queries the 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 search start point.

Syntax
```
:SEARch:SPOint {<NRf>}
:SEARch:SPOint?
```
<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:SEARCH:SPOINT -1
:SEARCH:SPOINT?
-> :SEARCH:SPOINT -1.000E+00
```

:SEARch:TWINdow

Function Sets or queries which window will display the detected points.

Syntax
```
:SEARch:TWINdow {Z1|Z2}
:SEARch:TWINdow?
```

Example
```
:SEARCH:TWINDOW Z1
:SEARCH:TWINDOW?
-> :SEARCH:TWINDOW Z1
```

:SEARch:TYPE

Function Sets or queries search type.

Syntax
```
:SEARch:TYPE {EDGE|QUALify|PWIDth|STA
Te|SWIDth}
:SEARch:TYPE?
```

Example
```
:SEARCH:TYPE EDGE
:SEARCH:TYPE?
-> :SEARCH:TYPE EDGE
```

:SEARch:WIDTh?

Function Queries all pulse width search settings.

Syntax `:SEARch:WIDTh?`

:SEARch:WIDTh:HYSTeresis

Function Sets or queries the pulse width search level hysteresis.

Syntax
```
:SEARch:WIDTh:HYSTeresis {<NRf>}
:SEARch:WIDTh:HYSTeresis?
```
<NRf> = 0.3 to 4 divisions in 0.1 steps

Example
```
:SEARCH:WIDTH:HYSTERESIS 1
:SEARCH:WIDTH:HYSTERESIS?
-> :SEARCH:WIDTH:HYSTERESIS 1.0
```

:SEARch:WIDTh:LEVel

Function Sets or queries the pulse width search level.

Syntax
```
:SEARch:WIDTh:LEVel {<NRf>|<Voltage>}
:SEARch:WIDTh:LEVel?
```
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SEARCH:WIDTH:LEVEL 1V
:SEARCH:WIDTH:LEVEL?
-> :SEARCH:WIDTH:LEVEL 1.000E+00
```

Description When the probe type is set to current, this command sets or queries the <Current> value.

:SEARch:WIDTh:POLarity

Function Sets or queries the pulse width search level polarity.

Syntax
```
:SEARch:WIDTh:POLarity {HIGH|LOW}
:SEARch:WIDTh:POLarity?
```

Example
```
:SEARCH:WIDTH:POLARITY HIGH
:SEARCH:WIDTH:POLARITY?
-> :SEARCH:WIDTH:POLARITY HIGH
```

:SEARch:WIDTh:SOURce

Function Sets or queries the pulse width search source waveform.

Syntax
```
:SEARch:WIDTh:SOURce {<NRf>|BIT<x>|
MATH<x>|NONE}
:SEARch:WIDTh:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> of BIT<x> = 1 to 8
<x> of MATH<x> = 1 or 2 (1 only for 2-channel models)

Example
```
:SEARCH:WIDTH:SOURCE 1
:SEARCH:WIDTH:SOURCE?
-> :SEARCH:WIDTH:SOURCE 1
```

:SEARch:WIDTh:TIME<x>

Function Sets or queries the pulse width for pulse width searching.

Syntax
```
:SEARch:WIDTh:TIME<x> {<Time>}
:SEARch:WIDTh:TIME<x>?
```
<x> = 1 or 2
<Time> = 1 ns to 500 s

Example
```
:SEARCH:WIDTH:TIME1 1S
:SEARCH:WIDTH:TIME1?
-> :SEARCH:WIDTH:TIME1 1.000E+00
```

:SEARch:WIDTh:TYPE

Function Sets or queries the pulse width search type.

Syntax
```
:SEARch:WIDTh:TYPE {BETWeen|IN|NOTBet
ween|OUT|TIMeout}
:SEARch:WIDTh:TYPE?
```

Example
```
:SEARCH:WIDTH:TYPE BETWEEN
:SEARCH:WIDTH:TYPE?
-> :SEARCH:WIDTH:TYPE BETWEEN
```

5.24 SERialbus Group

The following table indicates how the communication commands for serial bus types correspond to the menu items that appear on the screen.

Serial Bus Type	Communication Command	Setup Menu	
		Serial Bus Menu	Serial Bus Type
CAN bus signal	CAN*	SERIAL BUS	CAN
CAN FD bus signal	CANFD*		CAN FD
LIN bus signal	LIN*		LIN
CXPI bus signal	CXPI*		CXPI
SENT signal	SENT*		SENT
PSI5 signal	PSI5*		PSI5 Airbag
UART signal	UART*		UART
I2C bus signal	I2C*		I2C
SPI bus signal	SPI*		SPI
FlexRay bus signal	FLEXray*		FlexRay
User-defined serial bus signal	SPATtern		User Define

* CAN, CANFD, LIN, CXPI, SENT, PSI5, UART, I2C, SPI, and FLEXray are serial bus options available only for 4-channel models. Communication commands for these serial buses cannot be used on 2-channel models or 4-channel models without serial bus options.

:SERialbus<x>?

Function Queries all serial bus signal analysis and search settings.

Syntax `:SERialbus<x>?`
<x> = 1 to 4 (1 only for 2-channel models)

:SERialbus<x>:ASETup:ABORt

Function Aborts auto setup on a serial bus signal.

Syntax `:SERialbus<x>:ASETup:ABORt`
<x> = 1 to 4

Example `:SERIALBUS1:ASETUP:ABORT`

Description This command cannot be used on the DLM2022, DLM2032, or DLM2052.

:SERialbus<x>:ASETup:EXECute

Function Executes auto setup on a serial bus signal.

Syntax `:SERialbus<x>:ASETup:EXECute`
<x> = 1 to 4

Example `:SERIALBUS1:ASETUP:EXECUTE`

Description This command cannot be used on the DLM2022, DLM2032, or DLM2052.

: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 DLM2000 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 or 2

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 of 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 DLM2000 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 DLM2000 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 DLM2000 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:C
RC ON
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:C
RC?
-> :SERIALBUS1:CAN:SEARCH:SETUP:EFRAM
E: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:S
TUFF ON
:SERIALBUS1:CAN:SEARCH:SETUP:EFRAME:S
TUFF?
-> :SERIALBUS1:CAN:SEARCH:SETUP:EFRAM
E: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 ACK settings for the ID and data conditions 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[:IDDat
a]:ACK:MODE {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:ACK:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:A
CK:MODE ON
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:A
CK:MODE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A: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[:IDDat
a]:ACK:TYPE {ACK|ACKBoth|NONack}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:ACK:TYPE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:A
CK:TYPE ACK
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:A
CK:TYPE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A: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[:IDDat
a]:DATA?
```
<x> = 1 to 4

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:CONDition

Function Sets or queries the comparison condition, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:CONDition {BETWeen|EQUal|FALS
e|GREater|LESS|NOTBetween|NOTEqul|TR
UE}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:CONDITION BETWEEN
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:CONDITION?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A:DATA:CONDITION BETWEEN
```

: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[:IDDat
a]:DATA:DECimal<y> {<NRf>,<NRf>}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:DECIMAL1 1
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:DECIMAL1?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A:DATA:DECIMAL1 1
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:DLC

Function Sets or queries the data length code (DLC), which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:DLC {<NRf>}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:DLC?
```
<x> = 1 to 4
<NRf> = 0 to 8

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:DLC 0
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:DLC?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A:DATA:DLC 0
```

: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[:IDDat
a]:DATA:ENDian {BIG|LITTle}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:ENDian?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:ENDIAN BIG
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:ENDIAN?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A:DATA:ENDIAN BIG
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:HEXa<y>

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[:IDDat
a]:DATA:HEXa<y> {<String>}
```
<x> = 1 to 4
<y> = 1 to 8

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:HEXA1 "12"
```

:SERialbus<x>:CAN: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 bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:MODE {<Boolean>}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:MODE ON
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:MODE?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A: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[:IDDat
a]:DATA:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:CAN:SEARch:SETup[:IDDat
a]:DATA:MSBLsb?
```
<x> = 1 to 4
<NRf> = 0 to 63

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:MSBLSB 1,0
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:D
ATA:MSBLSB?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDAT
A:DATA:MSBLSB 1,0
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:PATTern<y>

Function Sets or queries the data value, 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:PATTern<y> {<String>}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:PATTern<y>?
```
<x> = 1 to 4
<y> = 1 to 8

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:PATTERN1 "00110101"
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:PATTERN1?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:PATTERN1 "00110101"
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:PFORmat

Function Sets or queries the data input format, which is one of the ID and data conditions, for CAN bus signal searching.

Syntax
```
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:PFORmat {BINary|HEXa}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:PFORMAT BINARY
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:PFORMAT?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:DATA:PFORMAT BINARY
```

:SERialbus<x>:CAN: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 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 all identifier settings for 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 the ID 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]:IDENtifier:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:HEXA "1AB"
```

: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 enable/disable condition, 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 pattern, 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"
```

: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 {BINary|HEXa|MESSage}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:IDENtifier:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT BINARY
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:IDENTIFIER:PFORMAT BINARY
```

:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal?

Function Queries all message and signal settings for 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 and 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 all signal settings for 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 {BETWeen|EQUal|GREater|LESS|NOTBetween|NOTEqual}
:SERialbus<x>:CAN:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION BETWEEN
:SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION?
-> :SERIALBUS1:CAN:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION BETWEEN
```

:SERialbus<x>:CAN: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 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 DLM2000 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.

Function
```
: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 analysis is compliant to the ISO standard.

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 or 2

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:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:CANFD:DETail:DISPlay
<x> = 1 to 4

Example :SERIALBUS1:CANFD:DETAIL:DISPLAY FULL
:SERIALBUS1:CANFD:DETAIL:DISPLAY?
-> :SERIALBUS1:CANFD:DETAIL:DISPLAY 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:ITEM?
<x> = 1 to 4

Example :SERIALBUS1:CANFD:DETAIL:LIST:ITEM?
-> :SERIALBUS1:CANFD:DETAIL:LIST:ITEM "No.,Time(ms),Frame,ID,DLC,Data,CRC,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:VALue? {<NRf>|MAXimum|MINimum}
<x> = 1 to 4
<NRf> = −49999 to 49999

Example :SERIALBUS1:CANFD:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:CANFD:DETAIL:LIST:VALUE "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 ACK Field in the CAN FD bus signal search result.

Syntax :SERialbus<x>:CANFD:SEARch:FJUMp:ACK
<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:CONTrol
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:CONTROL

: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:DATA
<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:IDENtifier
<x> = 1 to 4

Example :SERIALBUS1:CANFD:SEARCH:FJUMP:IDENTIFIER

: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 DLM2000 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 DLM2000 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>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:BCOunt?
```
<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 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 {BETWeen|EQUal|FALSe|GREater|LESS|NOTBetween|NOTEqul|TRUE}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:CONDITION BETWEEN
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:CONDITION BETWEEN
```

: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>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:DBYTe?
```
<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: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:HEXa<y>

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:HEXa<y> {<String>}
```
<x> = 1 to 4
<y> = 1 to 8

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:DATA:HEXA1 "12"
```

: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[:IDD
ata]:DATA:MODE {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]: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:IDD
ATA: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[:IDD
ata]:DATA:MSBLsb {<NRf>,<NRf>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]: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:IDD
ATA:DATA:MSBLSB 15,0
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:PATTern<y>

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[:IDD
ata]:DATA:PATTern<y> {<String>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]:DATA:PATTern<y>?
```
<x> = 1 to 4
<y> = 1 to 8

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:DATA:PATTERN1 "00110101"
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:DATA:PATTERN1?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDD
ATA:DATA:PATTERN1 "00110101"
```

:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:DATA:PFORmat

Function Sets or queries the data input format, which is one of the ID and data conditions, for CAN FD bus signal searching.

Syntax
```
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]:DATA:PFORmat {BINary|HEXa}
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA
:DATA:PFORMAT BINARY
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA
:DATA:PFORMAT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDD
ATA:DATA:PFORMAT BINARY
```

: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[:IDD
ata]:DATA:SIGN {SIGN|UNSign}
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]: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:IDD
ATA: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[:IDD
ata]: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[:IDD
ata]: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:IDD
ATA: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[:IDD
ata]:IDENtifier:MODE {<Boolean>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]: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:IDD
ATA: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[:IDD
ata]:IDENtifier:PATTern {<String>}
:SERialbus<x>:CANFD:SEARch:SETup[:IDD
ata]:IDENtifier:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:IDENTIFIER:PATTERN "1110000111
1"
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:IDENTIFIER:PATTERN?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDD
ATA: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 {BINary|HE
Xa|MESSage}
:SERialbus<x>:CANFD:SEARch:SETup[:
IDData]:IDENtifier:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:IDENTIFIER:PFORMAT BINARY
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:IDENTIFIER:PFORMAT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDD
ATA:IDENTIFIER:PFORMAT BINARY
```

: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[:IDD
ata]: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[:IDD
ata]: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:SELect {MESSage|SIGNa
l}
:SERialbus<x>:CANFD:SEARch:SETup[:
IDData]:MSIGnal:SELect?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:MSIGNAL:SELECT MESSAGE
:SERIALBUS1:CANFD:SEARCH:SETUP:
IDDATA:MSIGNAL:SELECT?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDD
ATA: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[:IDData]: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[:IDData]:MSIGnal:SIGNal:CONDition {BETWeen|EQUal|GREater|LESS|NOTBetween|NOTEqual}`

`:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:MSIGnal:SIGNal:CONDition?`

<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION BETWEEN
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:CONDITION BETWEEN
```

: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[:IDData]:MSIGnal:SIGNal:DECimal<y>?`

<x> = 1 to 4

<y> = 1 or 2

<NRf> = –9E+18 to 9E+18

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:DECIMAL1 1000
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:DECIMAL1?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:DECIMAL1 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:SETup[:IDData]:MSIGnal:SIGNal:ITEM {<String>,<String>}`

<x> = 1 to 4

<String> = Up to 32 characters

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:MSIGNAL:SIGNAL:ITEM "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[:IDData]:RTR {<Boolean>}`

`:SERialbus<x>:CANFD:SEARch:SETup[:IDData]:RTR?`

<x> = 1 to 4

Example
```
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:RTR ON
:SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA:RTR?
-> :SERIALBUS1:CANFD:SEARCH:SETUP:IDDATA: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 bus 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 ±10.0% (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 2
<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 DLM2000 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 49999

Example
```
:SERIALBUS1:CXPI:SEARCH:SELECT 1
:SERIALBUS1:CXPI:SEARCH:SELECT?
-> :SERIALBUS1:CXPI:SEARCH:
SELECT 1.50000000
```

Description If there are no detected points, the DLM2000 returns "NAN."

:SERialbus<x>:CXPI:SEARch:SELect? MAXimum

Function Queries the number of detected points in the CXPI bus signal search.

Syntax
```
:SERialbus<x>:CXPI:SEARch:
SELect? {MAXimum}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:
SELECT? MAXIMUM
-> :SERIALBUS1:CXPI:SEARCH:
SELECT 100
```

Description If there are no detected points, the DLM2000 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:C
LOCK ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:C
LOCK?
-> :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:C
OUNTER ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:C
OUNTER?
-> :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:C
RC ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:C
RC?
-> :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:D
LENGTH ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:D
LENGTH?
-> :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:F
RAMING ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:F
RAMING?
-> :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:I
BS ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:I
BS?
-> :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:P
ARITY ON
:SERIALBUS1:CXPI:SEARCH:SETUP:ERROR:P
ARITY?
-> :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 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 {BETWeen|EQUal|FALSe|
GREater|LESS|NOTBetween|NOTEqul|
TRUE}
:SERialbus<x>:CXPI:SEARch:
SETup[:IDData]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:CONDITION BETWEEN
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:CONDITION?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:CONDITION BETWEEN
```

: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, 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: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:HEXa<y>

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:
HEXa<y> {<String>}
:SERialbus<x>:CXPI:SEARch:
SETup[:IDData]:DATA:HEXa<y>?
```
<x> = 1 to 4
<y> = 1 to 8

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:HEXA1 "12"
```

: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:PATTern<y>

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:
PATTern<y> {<String>}
:SERialbus<x>:CXPI:SEARch:
SETup[:IDData]:DATA:PATTern<y>?
```
<x> = 1 to 4
<y> = 1 to 8

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:PATTERN1 "00110101"
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:PATTERN1?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:PATTERN1 "00110101"
```

:SERialbus<x>:CXPI:SEARch:SETup[:IDData]:DATA:PFORmat

Function Sets or queries the data input format, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:
SETup[:IDData]:DATA:
PFORmat {BINary|HEXa}
:SERialbus<x>:CXPI:SEARch:
SETup[:IDData]:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:PFORMAT BINARY
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:PFORMAT?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:DATA:PFORMAT BINARY
```

:SERialbus<x>:CXPI: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 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:PFORmat

Function Sets or queries the ID input format, which is one of the ID and data conditions, for CXPI bus signal searching.

Syntax
```
:SERialbus<x>:CXPI:SEARch:
SETup[:IDData]:ID:
PFORmat {BINary|HEXa}
:SERialbus<x>:CXPI:SEARch:
SETup[:IDData]:ID:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:ID:PFORMAT BINARY
:SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:ID:PFORMAT?
-> :SERIALBUS1:CXPI:SEARCH:SETUP:
IDDATA:ID:PFORMAT BINARY
```

: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:WAKeupsleep?

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:WAKeupsleep: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:WAKeupsleep: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:WAKeupsleep: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:WAKeupsleep: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 (1 only for 2-channel models)

:SERialbus<x>:DECode[:FORMat]

Function Sets or queries the decode display format of serial bus signals.

Syntax
```
:SERialbus<x>:DECode[:FORMat]{BINary|
HEXa|ASCii|SYMBol}
:SERialbus<x>:DECode[:FORMat]?
```
<x> = 1 to 4 (1 only for 2-channel models)

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 I^2C serial bus signals.

Syntax
```
:SERialbus<x>:DECode:SSCMode {<Boolea
n>}
:SERialbus<x>:DECode:SSCMode?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:DECODE:SSCMODE ON
:SERIALBUS1:DECODE:SSCMODE?
-> :SERIALBUS1:DECODE:SSCMODE 1
```

:SERialbus<x>:DISPlay

Function Sets or queries whether or not serial bus signal analysis will be performed.

Syntax
```
:SERialbus<x>:DISPlay {<Boolean>}
:SERialbus<x>:DISPlay?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:DISPLAY ON
:SERIALBUS1:DISPLAY?
-> :SERIALBUS1:DISPLAY 1
```

:SERialbus<x>:FLEXray?

Function Queries all FLEXRAY bus signal settings.

Syntax
```
:SERialbus<x>:FLEXray?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:ANALyze?

Function Queries all FLEXRAY bus signal settings.

Syntax
```
:SERialbus<x>:FLEXray:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray[:ANALyze]:SETup?

Function Queries all FLEXRAY bus signal analysis settings.

Syntax
```
:SERialbus<x>:FLEXray[:ANALyze]:SET
up?
```
<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:BCH
ANNEL A
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BCH
ANNEL?
-> :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, or 10000000

Example
```
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BRA
TE 5000000
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:BRA
TE?
-> :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:
BRATE 5000000
```

:SERialbus<x>:FLEXray[:ANALyze]:SETup:SOURce

Function Sets or queries the source signal for FLEXRAY bus signal analysis.

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 or 2

Example
```
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:SOU
RCE 1
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:SOU
RCE?
-> :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:
SOURCE 1
```

:SERialbus<x>:FLEXray[:ANALyze]:SETup:SPOint

Function Sets or queries the FLEXRAY bus signal sample point.

Syntax
```
:SERialbus<x>:FLEXray[:ANALyze]:
SETup:SPOint {<NRf>}
:SERialbus<x>:FLEXray[:ANALyze]:
SETup:SPOint?
```
<x> = 1 to 4

<NRf> = 4, 5, or 6

Example
```
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:SPO
INT 5
:SERIALBUS1:FLEXRAY:ANALYZE:SETUP:SPO
INT?
-> :SERIALBUS1:FLEXRAY:ANALYZE:SETUP:
SPOINT 5
```

: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:DISPl
ay {FULL|LOWer|UPPer}
:SERialbus<x>:FLEXray:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:DETAIL:DISPL
AY FULL
:SERIALBUS1:FLEXRAY:DETAIL:DISPLAY?
-> :SERIALBUS1:FLEXRAY:DETAIL:DISPL
AY 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:IT
EM?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:DETAIL:LIST:ITEM?
-> :SERIALBUS1:FLEXRAY:DETAIL:LIST:IT
EM "No.,Time(ms),S/D,IND,ID,Len,CC,Da
ta,Information,"
```

:SERialbus<x>:FLEXray:DETail:LIST:VALue?

Function Queries all of the data for the specified analysis number in the FLEXRAY bus signal analysis result list.

Syntax
```
:SERialbus<x>:FLEXray:DETail:LIST:VAL
ue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4

<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:FLEXRAY:DETAIL:LIST:VAL
UE? 0
-> :SERIALBUS1:FLEXRAY:DETAIL:LIST:V
ALUE "     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 analysis results.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:FJUMp:CC
Ount
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:FJUMP:CCOU
NT
```

:SERialbus<x>:FLEXray:SEARch:FJUMp:CRC

Function Jumps to the CRC Field in the FLEXRAY bus signal analysis results.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:FJUMp:C
RC
```
<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 analysis results.

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 analysis results.

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 analysis results.

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 of the FLEXRAY bus signal search, or queries the zoom location that corresponds to the 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 searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup:IDData?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt?

Function Queries all cycle-count settings for FLEXRAY bus signal searching.

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 cycle-count data-condition settings for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition {BETWeen|EQUal|GREater|LESS|NOTBetween|NOTEqul}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:CONDITION BETWEEN
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:CONDITION?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:CCOUNT:CONDITION BETWEEN
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt<y>

Function Sets or queries a 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
- For `:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition EQUal|GREater|NOTEqul`, set the cycle-count setting using `:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt1`.
- For `:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition LESS`, set the cycle-count setting using `:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt2`.
- For `:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:CONDition BETWeen|NOTBetween`, set the smaller value using `:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt1`, and set the larger value using `:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:CCOunt:COUNt2`.

: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 all ID and data condition settings 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 position for comparing data patterns, 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 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 {BETWeen|EQUal|FALSe|GREater|LESS|NOTBetween|NOTEqul|TRUE}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:CONDITION BETWEEN
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:CONDITION BETWEEN
```

: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 DLM2000 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: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:HEXa<y>

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:HEXa<y> {<String>}
```
<x> = 1 to 4
<y> = 1 to 8
<String> = Two characters (each representing a byte). Each character can be X or a hexadecimal value from 0 to F.

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:HEXA1 "1F"
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:MODE

Function Sets or queries the data enable/disable condition, 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:PATTern<y>

Function Sets or queries the data value, 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:PATTern<y> {<String>}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:PATTern<y>?
```
<x> = 1 to 4
<y> = 1 to 8
<String> = Eight characters (each representing a byte). Each character can be 0, 1, or X.

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:PATTERN1 "00001010"
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:PATTERN1?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:PATTERN1 "00001010"
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:PFORmat

Function Sets or queries the data input format, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:PFORmat {BINary|HEXa}
:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:PFORMAT BINARY
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:PFORMAT?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDATA:DATA:PFORMAT BINARY
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:DATA:SIGN

Function Sets or queries whether a signed or unsigned data format will be used (this is one of the ID and data conditions) for CAN 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 all frame ID settings for the ID and data conditions for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]: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[:I
DData]:FID:CONDition {BETWeen|EQUal|G
REater|LESS|NOTBetween|NOTEqul}
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:FID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:FID:CONDITION BETWEEN
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:FID:CONDITION?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:I
DDATA:FID:CONDITION BETWEEN
```

: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[:I
DData]:FID:ID<y> {<NRf>}
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:FID:ID<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 to 2047

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:FID:ID1 100
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:FID:ID1?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:I
DDATA:FID:ID1 100
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:FID:MODE

Function Sets or queries the frame ID enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:FID:MODE {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:FID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:FID:MODE ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:FID:MODE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:I
DDATA:FID:MODE 1
```

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator?

Function Queries all indicator settings for the ID and data conditions for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:INDIcator?
```
<x> = 1 to 4

:SERialbus<x>:FLEXray:SEARch:SETup[:IDData]:INDIcator:MODE

Function Sets or queries the indicator enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal searching.

Syntax
```
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:INDIcator:MODE {<Boolean>}
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:INDIcator:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:INDICATOR:MODE ON
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:INDICATOR:MODE?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:I
DDATA: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[:I
DData]:INDIcator:NFRame {0|1|X}
:SERialbus<x>:FLEXray:SEARch:SETup[:I
DData]:INDIcator:NFRame?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:INDICATOR:NFRAME 1
:SERIALBUS1:FLEXRAY:SEARCH:SETUP:IDDA
TA:INDICATOR:NFRAME?
-> :SERIALBUS1:FLEXRAY:SEARCH:SETUP:I
DDATA: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 whether or not 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.

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 I^{2}C bus signal analysis and search settings.

Syntax
```
:SERialbus<x>:I2C?
```
<x> = 1 to 4

:SERialbus<x>:I2C:ANALyze?

Function Queries all I^{2}C bus signal analysis settings.

Syntax
```
:SERialbus<x>:I2C:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:I2C[:ANALyze]:SETup?

Function Queries all I^{2}C bus signal analysis bus settings.

Syntax
```
:SERialbus<x>:I2C[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:I2C[:ANALyze]:SETup:CLOCk?

Function Queries all I^{2}C 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 clock source for I^{2}C bus signal analysis.

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 or 2

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 I^{2}C 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 I^{2}C bus signal data source.

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 or 2

Example
```
:SERIALBUS1:I2C:ANALYZE:SETUP:DATA:SOURCE 1
:SERIALBUS1:I2C:ANALYZE:SETUP:DATA:SOURCE?
-> :SERIALBUS1:I2C:ANALYZE:SETUP:DATA:SOURCE 1
```

:SERialbus<x>:I2C:DETail?

Function Queries all I^{2}C 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 I^{2}C 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 I^{2}C 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 of the data for the specified analysis number in the I^{2}C bus signal analysis result list.

Syntax
```
:SERialbus<x>:I2C:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4
<NRf> = See the DLM2000 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[:ANALyze]:SETup:INCLuderw

Function Sets or queries the on/off status of the R/W address bit in I^{2}C 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:SEARch?

Function Queries all I^{2}C bus signal search settings.

Syntax
```
:SERialbus<x>:I2C:SEARch?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch:ABORt

Function Aborts the I^{2}C 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 a I^{2}C 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 I^{2}C 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 DLM2000 returns “NAN.”

:SERialbus<x>:I2C:SEARch:SELect? MAXimum

Function Queries the number of detected points in the I^{2}C bus signal search.

Syntax
```
:SERialbus<x>:I2C:SEARch:SELe
ct? {MAXimum}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SELECT? MAXIMU
M
-> :SERIALBUS1:I2C:SEARCH:SELECT 100
```

Description If there are no detected points, the DLM2000 returns “NAN.”

:SERialbus<x>:I2C:SEARch:SETup?

Function Queries all I^{2}C bus signal search condition settings.

Syntax
```
:SERialbus<x>:I2C:SEARch:SETup?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess?

Function Queries all address pattern settings for I^{2}C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDR
ess?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess?

Function Queries all address pattern address settings for I^{2}C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDR
ess:ADDRess?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress?

Function Queries all 10-bit address settings for I^{2}C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDR
ess:ADDRess:BIT10ADdress?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:BIT10ADdress:HEXa

Function Sets the 10-bit address for I^{2}C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDR
ess: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 I^{2}C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDR
ess:ADDRess:BIT10ADdress:
PATTern {<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDR
ess:ADDRess:BIT10ADdress:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADD
RESS:ADDRESS:BIT10ADDRESS:PATTE
RN "11010111001"
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:
ADDRESS:BIT10ADDRESS:PATTERN?
-> :SERIALBUS1:I2C:SEARCH:SETUP:A
DDRESS:ADDRESS:BIT10ADDRESS:PATTE
RN "11010111001"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]?

Function Queries all 7-bit address settings for I^{2}C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDR
ess:ADDRess[:BIT7ADdress]?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess[:BIT7ADdress]:HEXa

Function Sets the 7-bit address for I^2C 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 I^2C 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 7-bit + sub address settings for I^2C bus signal searching.

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 7-bit address settings for the 7-bit + sub address for I^2C 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:HEXa

Function Sets the 7-bit address for the 7-bit + sub address for I^2C 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 for the 7-bit + sub address for I^2C 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 sub address settings for the 7-bit + sub address for I^2C 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 for the 7-bit + sub address for I^2C 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 for the 7-bit + sub address for I^{2}C 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 mode for I^{2}C 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:PFORmat

Function Sets or queries the address input format, which is one of the address conditions, for I^{2}C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:PFORmat {BINary|HEXa}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:PFORMAT BINARY
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:PFORMAT?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:ADDRESS:PFORMAT BINARY
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:ADDRess:TYPE

Function Sets or queries the address type, which is one of the address conditions, for I^{2}C 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 address pattern data settings for I^{2}C bus signal searching.

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 position for comparing data patterns for I^{2}C 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 the on/off status of the position for comparing data patterns for I^{2}C 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 comparison condition (true or false) for I^2C 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 data bytes that will be compared for I^2C 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:HEXa<y>

Function Sets the data value, which is one of the data conditions, for I^2C bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:HEXa<y> {<String>}
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:HEXA1 "AB"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:MODE

Function Sets or queries the data condition enable/disable mode for I^2C 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]:ADDRess:DATA:PATTern<y>

Function Sets or queries the data value, which is one of the data conditions, for I^2C bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:PATTern<y> {<String>}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:PATTern<y>?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:PATTERN1 "10001101"
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:PATTERN1?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:PATTERN1 "10001101"
```

:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:PFORmat

Function Sets or queries the data input format, which is one of the data conditions, for I^2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:PFORmat {BINary|HEXa}
:SERialbus<x>:I2C:SEARch[:SETup]:ADDRess:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:PFORMAT BINARY
:SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:PFORMAT?
-> :SERIALBUS1:I2C:SEARCH:SETUP:ADDRESS:DATA:PFORMAT BINARY
```

:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall?

Function Queries all general call settings for I^2C bus signal searching.

Syntax
```
:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:GENeralcall:BIT7Maddress?

Function: Queries all general call's 7-bit master address settings for I^{2}C 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 general call's 7-bit master address for I^{2}C 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 general call's 7-bit master address for I^{2}C 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's second byte type for I^{2}C 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 I^{2}C bus signal search type.

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 NON ACK ignore mode settings for I^{2}C bus signal searching.

Syntax:
```
:SERialbus<x>:I2C:SEARch[:SETup]:NONack?
```
<x> = 1 to 4

:SERialbus<x>:I2C:SEARch[:SETup]:NONack:HSMode

Function: Sets or queries whether or not NON ACK will be ignored in high-speed mode for I^{2}C 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 or not NON ACK will be ignored in read access mode for I^{2}C 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 or not NON ACK will be ignored in start bytes for 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:S
TARTBYTE ON
:SERIALBUS1:I2C:SEARCH:SETUP:NONACK:S
TARTBYTE?
-> :SERIALBUS1:I2C:SEARCH:SETUP:NONAC
K: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:BRA
Te {<NRf>|USER,<NRf>}
:SERialbus<x>:LIN[:ANALyze]:SETup:BRA
Te?
```
<x> = 1 to 4
<NRf> = 1200, 2400, 4800, 9600, 19200
USER <NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:BRA
TE 2400
:SERIALBUS1:LIN:ANALYZE:SETUP:BRATE?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:BRA
TE 2400
```

:SERialbus<x>:LIN[:ANALyze]:SETup:REVision

Function Sets or queries the LIN bus signal analysis revision number.

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup:REV
ision {BOTH|LIN1_3|LIN2_0}
:SERialbus<x>:LIN[:ANALyze]:SETup:REV
ision?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:REVISI
ON LIN1_3
:SERIALBUS1:LIN:ANALYZE:SETUP:REVISI
ON?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:REVI
SION LIN1_3
```

:SERialbus<x>:LIN[:ANALyze]:SETup:SOURce

Function Sets or queries the LIN bus signal analysis source.

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup:SOU
Rce {<NRf>|MATH<y>}
:SERialbus<x>:LIN[:ANALyze]:SETup:SOU
Rce?
```
<x> = 1 to 4
<NRf> = 1 to 4
<y> of MATH<y> = 1 or 2

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:SOUR
CE 1
:SERIALBUS1:LIN:ANALYZE:SETUP:SOURCE?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:SOUR
CE 1
```

:SERialbus<x>:LIN[:ANALyze]:SETup:SPOint

Function Sets or queries the LIN bus signal sample point.

Syntax
```
:SERialbus<x>:LIN[:ANALyze]:SETup:SPO
int {<NRf>}
:SERialbus<x>:LIN[:ANALyze]:SETup:SPO
int?
```
<x> = 1 to 4
<NRf> = 18.8 to 90.6

Example
```
:SERIALBUS1:LIN:ANALYZE:SETUP:SPOI
NT 18.8
:SERIALBUS1:LIN:ANALYZE:SETUP:SPOINT?
-> :SERIALBUS1:LIN:ANALYZE:SETUP:SPOI
NT 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 FULL
```

: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:LIST:
ITEM "No.,Time(ms),ID,ID-Field,Data,
Checksum,Information"
```

:SERialbus<x>:LIN:DETail:LIST:VALue?

Function Queries all of the data for the specified analysis number in the LIN bus signal analysis result list.

Syntax
```
:SERialbus<x>:LIN:DETail:LIST:
VALue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:LIN:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:LIN:DETAIL:LIST:
VALUE "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:IDENti
fier
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:FJUMP:IDENTIFI
ER
```

: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 {<NRf
>|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 1.50000000
```

Description If there are no detected points, the DLM2000 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 DLM2000 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:CH
ECKSUM ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:CH
ECKSUM?
-> :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:FR
AMING ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:FR
AMING?
-> :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:PA
RITY ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:PA
RITY?
-> :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:SY
NCH ON
:SERIALBUS1:LIN:SEARCH:SETUP:ERROR:SY
NCH?
-> :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 data conditions for LIN bus signal searching.

Syntax:
```
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:CONDition {BETWeen|EQUal|FALSe|GREater|LESS|NOTBetween|NOTEqul|TRUE}
:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:CONDition?
```
<x> = 1 to 4

Example:
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:CONDITION BETWEEN
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:CONDITION?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:DATA:CONDITION BETWEEN
```

: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 value 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 DLM2000 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:ENDian

Function: Sets or queries the data endian setting 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:HEXa<y>

Function Sets the data for LIN bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData
:DATA:HEXa<y> {<String>}
```
<x> = 1 to 4
<y> = 1 to 8

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:HEXA1 "12"
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MODE

Function Sets or queries the data enable/disable condition 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:D
ATA:MODE ON
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:MODE?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:DATA:MODE 1
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:MSBLsb

Function Sets or queries the data MSB and LSB bits 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:D
ATA:MSBLSB 1,0
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:MSBLSB?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:DATA:MSBLSB 1,0
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:PATTern<y>

Function Sets or queries the data value for LIN bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDData
:DATA:PATTern<y> {<String>}
:SERialbus<x>:LIN:SEARch:SETup:IDData
:DATA:PATTern<y>?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:PATTERN1 "00110101"
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:PATTERN1?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:DATA:PATTERN1 "00110101"
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:PFORmat

Function Sets or queries the input format, which is one of the data conditions, for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:
IDData:DATA:PFORmat {BINary|HEXa}
:SERialbus<x>:LIN:SEARch:SETup:
IDData:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:PFORMAT BINARY
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:PFORMAT?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:DATA:PFORMAT BINARY
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:DATA:SIGN

Function Sets or queries whether signed or unsigned data format will be used, which is one of the data conditions, 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:D
ATA:SIGN SIGN
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:D
ATA:SIGN?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:DATA:SIGN SIGN
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier?

Function Queries all identifier settings for the ID and data conditions for LIN bus signal searching.

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:I
DENTIFIER:ID:HEXA "1E"
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:MODE

Function Sets or queries the ID enable/disable condition 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:I
DENTIFIER:ID:MODE ON
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:I
DENTIFIER:ID:MODE?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:IDENTIFIER:ID:MODE 1
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:ID:PATTern

Function Sets or queries the ID value for LIN bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:IDDat
a:IDENtifier:ID:PATTern {<String>}
:SERialbus<x>:LIN:SEARch:SETup:IDDat
a:IDENtifier:ID:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:I
DENTIFIER:ID:PATTERN "101100"
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:I
DENTIFIER:ID:PATTERN?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:IDENTIFIER:ID:PATTERN "101100"
```

:SERialbus<x>:LIN:SEARch:SETup:IDData:IDENtifier:PFORmat

Function Sets or queries the ID input format for LIN bus signal searching.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:
IDData:IDENtifier:PFORmat {BINary|HEX
a}
:SERialbus<x>:LIN:SEARch:SETup:
IDData:IDENtifier:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:I
DENTIFIER:PFORMAT BINARY
:SERIALBUS1:LIN:SEARCH:SETUP:IDDATA:I
DENTIFIER:PFORMAT?
-> :SERIALBUS1:LIN:SEARCH:SETUP:IDDAT
A:IDENTIFIER:PFORMAT BINARY
```

:SERialbus<x>:LIN:SEARch:SETup:MODE

Function Sets or queries the LIN bus signal search mode setting.

Syntax
```
:SERialbus<x>:LIN:SEARch:SETup:MO
DE {BSYNch|ERRor|IDData}
:SERialbus<x>:LIN:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:LIN:SEARCH:SETUP:MODE BSY
NCH
:SERIALBUS1:LIN:SEARCH:SETUP:MODE?
-> :SERIALBUS1:LIN:SEARCH:SETUP:
MODE BSYNCH
```

:SERialbus<x>:PSI5?

Function Queries all PSI5 signal analysis and search settings.

Syntax
```
:SERialbus<x>:PSI5?
```
<x> = 1 to 4

:SERialbus<x>:PSI5:ANALyze?

Function Queries all PSI5 signal analysis settings.

Syntax
```
:SERialbus<x>:PSI5:ANALyze?
```
<x> = 1 to 4

:SERialbus<x>:PSI5[:ANALyze]:SETup?

Function Queries all bus setup settings for PSI5 signal analysis.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup?
```
<x> = 1 to 4

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA?

Function Queries all PSI5 signal analysis data signal settings.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DA
TA?
```
<x> = 1 to 4

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:BRATe

Function Sets or queries the PSI5 signal analysis bit rate (data transfer rate).

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:BRATe {<NRf>|USER,<NRf>}
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:BRATe?
```
<x> = 1 to 4

<NRf> = 125000, 189000

USER <NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:BRATE 125000
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:BRATE?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:BRATE 125000
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:CTOLerance

Function Sets or queries the PSI5 signal analysis clock tolerance.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:CTOLerance {<NRf>}
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:CTOLerance?
```
<NRf> = ±0.5% to ±33.3% (resolution:0.1%)

<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:CTOLERANCE 5
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:CTOLERANCE?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:CTOLERANCE 5.00E+00
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:DBITs

Function Sets or queries the data length for PSI5 signal analysis.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:DBITs {<NRf>}
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:DBITs?
```
<x> = 1 to 4

<NRf> = 10, 16

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:DBITS 10
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:DBITS?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:DBITS 10
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:EDETection

Function Sets or queries the error detection method for PSI5 signal analysis.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:EDETection {CRC|PARity}
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:EDETection?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:EDETECTION CRC
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:EDETECTION?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:EDETECTION CRC
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection

Function Queries all PSI5 signal analysis noise rejection settings.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection?
```
<x> = 1 to 4

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:ETIMe

Function Sets or queries the rejection end time for PSI5 signal analysis noise rejection.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:ETIMe
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:ETIMe?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SNREJECTION:ETIME 66.0000E-06
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SNREJECTION:ETIME?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SNREJECTION:ETIME 66.0000E-6
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:MODE

Function Sets or queries the on/off status of the PSI5 signal analysis noise rejection.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:MODE
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SNRejection:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SNREJECTION:MODE 1
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SNREJECTION:MODE?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SNREJECTION:MODE 1
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SOURce

Function Sets or queries the data source for PSI5 signal analysis.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SOURce {<NRf>|MATH<y>}
:SERialbus<x>:PSI5[:ANALyze]:SETup:DATA:SOURce?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 to 4

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SOURCE 1
:SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SOURCE?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:DATA:SOURCE 1
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:NUMBerofslot

Function Sets or queries the number of slots for PSI5 signal analysis.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:NUMBerofslot {AUTO|<NRf>}
:SERialbus<x>:PSI5[:ANALyze]:SETup:NUMBerofslot?
```
<x> = 1 to 4
<NRf> = 1 to 6

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:NUMBEROFSLOT AUTO
:SERIALBUS1:PSI5:ANALYZE:SETUP:NUMBEROFSLOT?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:NUMBEROFSLOT AUTO
```

:SERialbus<x>:PSI5[:ANALyze]:SETup:SLOT<y>

Function Sets or queries the slot’s start and end times for PSI5 signal analysis.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:SLOT<y> {<NRf>[,<NRf>]}
:SERialbus<x>:PSI5[:ANALyze]:SETup:SLOT<y>?
```
<x> = 1 to 4
<y> = 1 to 6 (slot number)
<NRf> = 0 to 0.02 s (in 0.1 us steps)

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:SLOT6 870us,1088us
:SERIALBUS1:PSI5:ANALYZE:SETUP:SLOT1?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:SLOT1 870.000E-06,1.08800E-03
```

Description The first <NRF> is the slot's start time and the second the end time.

If :SERialbus<x>:PSI5[:ANALyze]:SETup:NUMBerofslot is a number from 1 to 6, when the last slot number is specified with <y>, the second <NRf> becomes valid.

If :SERialbus<x>:PSI5[:ANALyze]:SETup:NUMBerofslot is AUTO, query is not possible.

:SERialbus<x>:PSI5[:ANALyze]:SETup:SYNC

Function Sets or queries the sync signal for PSI5 signal analysis.

Syntax
```
:SERialbus<x>:PSI5[:ANALyze]:SETup:SYNC {<NRf>|MATH<y>|NONE}
:SERialbus<x>:PSI5[:ANALyze]:SETup:SYNC?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 to 4

Example
```
:SERIALBUS1:PSI5:ANALYZE:SETUP:SYNC 1
:SERIALBUS1:PSI5:ANALYZE:SETUP:SYNC?
-> :SERIALBUS1:PSI5:ANALYZE:SETUP:SYNC 1
```

:SERialbus<x>:PSI5:DETail?

Function Queries all PSI5 signal analysis list settings.

Syntax
```
:SERialbus<x>:PSI5:DETail?
```
<x> = 1 to 4

:SERialbus<x>:PSI5:DETail:DISPlay

Function Sets or queries the display mode for the PSI5 signal analysis list.

Syntax
```
:SERialbus<x>:PSI5:DETail:DISPlay {FULL|LOWer|UPPer}
:SERialbus<x>:PSI5:DETail:DISPlay?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:DETAIL:DISPLAY FULL
:SERIALBUS1:PSI5:DETAIL:DISPLAY?
-> :SERIALBUS1:PSI5:DETAIL:DISPLAY FULL
```

:SERialbus<x>:PSI5:DETail:LIST:ALL?

Function Queries all the data for all analysis numbers in the PSI5 signal analysis result list.

Syntax `:SERialbus<x>:PSI5:DETail:LIST:ALL?`
<x> = 1 to 4

Example :SERIALBUS1:PSI5:DETAIL:LIST:ALL?
-> #8 (number of bytes, 8 digits)(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>:PSI5:DETail:LIST:ITEM?

Function Queries all items that will be displayed in the PSI5 signal analysis result list.

Syntax `:SERialbus<x>:PSI5:DETail:LIST:ITEM?`
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:DETAIL:LIST:ITEM?
-> :SERIALBUS1:PSI5:DETAIL:LIST:IT
EM "No.Time(ms)Slot No.DataParity/
CRCInformation"
```

:SERialbus<x>:PSI5:DETail:LIST:VALue?

Function Queries all the data for the specified analysis number in the PSI5 signal analysis result list.

Syntax `:SERialbus<x>:PSI5:DETail:LIST:VALue? {<NRf>|MAXimum|MINimum}`
<x> = 1 to 4
<NRf> = −400000 to 400000

Example
```
:SERIALBUS1:PSI5:DETAIL:LIST:VALUE? 1
-> :SERIALBUS1:PSI5:DETAIL:LIST:VAL
UE "1, 0.062104,1,AB67,2,,,"
```

:SERialbus<x>:PSI5:SEARch?

Function Queries all PSI5 signal search settings.

Syntax `:SERialbus<x>:PSI5:SEARch?`
<x> = 1 to 4

:SERialbus<x>:PSI5:SEARch:ABORt

Function Aborts the PSI5 signal search.

Syntax `:SERialbus<x>:PSI5:SEARch:ABORt`
<x> = 1 to 4

Example `:SERIALBUS1:PSI5:SEARCH:ABORT`

:SERialbus<x>:PSI5:SEARch:EXECute

Function Executes a PSI5 signal search.

Syntax `:SERialbus<x>:PSI5:SEARch:EXECute`
<x> = 1 to 4

Example `:SERIALBUS1:PSI5:SEARCH:EXECUTE`

:SERialbus<x>:PSI5:SEARch:SELect

Function Sets which detected point to display in the PSI5 signal search zoom window and queries the zoom position of the detected point.

Syntax
```
:SERialbus<x>:PSI5:SEARch:SELe
ct {<NRf>|MAXimum}
:SERialbus<x>:PSI5:SEARch:SELect?
```
<x> = 1 to 4
<NRf> = 0 to 49999

Example
```
:SERIALBUS1:PSI5:SEARCH:SELECT 1
:SERIALBUS1:PSI5:SEARCH:SELECT?
-> :SERIALBUS1:PSI5:SEARCH:
SELECT 1.50000000
```

Description If there are no detected points, the DLM2000 returns “NAN.”

:SERialbus<x>:PSI5:SEARch:SETup?

Function Queries all PSI5 signal search condition settings.

Syntax `:SERialbus<x>:PSI5:SEARch:SETup?`
<x> = 1 to 4

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA?

Function Queries all PSI5 signal data search settings.

Syntax `:SERialbus<x>:PSI5:SEARch[:SETup]:DATA?`
<x> = 1 to 4

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA?

Function Queries all data settings of the PSI5 signal data search.

Syntax `:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA?`
<x> = 1 to 4

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:CONDition

Function Sets or queries the comparison condition of the PSI5 signal data search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:CONDition {BETWeen|EQUal|FA
LSe|GREater|LESS|NOTBetween|NOTEqul|T
RUE}
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:CONDITION BETWEEN
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:CONDITION?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:CONDITION BETWEEN
```

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:DECimal<y>

Function Sets or queries the data of the PSI5 signal data search in decimal notation.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:DECimal<y> {<NRf>}
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2
When <NRf> = 10 bit data, −512 to 511
16 bit data, −32768 to 32767

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:DATA:DA
TA:DECIMAL1 -10
:SERIALBUS1:PSI5:SEARCH:SETUP:DATA:DA
TA:DECIMAL1?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:DECIMAL1 -10
```

Description This command is valid when :SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:CONDition is BETWeen, EQUal, GREater, LESS, NOTBetween, or NOTEqul.

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:HEXa

Function Sets the data of the PSI5 signal data search in hexadecimal notation.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:DATA:DA
TA:HEXA "3A0F"
```

Description This command is valid when :SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:CONDition is TRUE or FALSe.

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:PATTern

Function Sets or queries the data of the PSI5 signal data search in binary notation.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:PATTern {<String>}
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:PATTERN "0011010100001111"
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:PATTERN?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:PATTERN "0011010100001111"
```

Description This command is valid when :SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:CONDition is TRUE or FALSe.

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:DATA:PFORmat

Function Sets or queries the data input format of the PSI5 signal data search in binary notation.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:PFORmat {BINary|HEXa}
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:PFORMAT BINARY
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:PFORMAT?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:DATA:PFORMAT BINARY
```

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:FRAMeinslot?

Function Queries all slot specification settings of the PSI5 signal data search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:FRAMeinslot?
```
<x> = 1 to 4

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:FRAMeinslot:MODE

Function Sets or queries the slot specification condition (enabled/disabled) of the PSI5 signal data search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:FRAMeinslot:MODE {<Boolean>}
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:FRAMeinslot:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:FRAMEINSLOT:MODE ON
:SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:FRAMEINSLOT:MODE?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:FRAMEINSLOT:MODE 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:DATA:FRAMeinslot:SNUMber

Function Sets or queries the slot number of the PSI5 signal data search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:FRAMeinslot:SNUMber {<NRf>}
:SERialbus<x>:PSI5:SEARch[:SETup]:
DATA:FRAMeinslot:SNUMber?
```
<x> = 1 to 4
<NRf> = 1 to 6

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:DATA:FR
AMEINSLOT:SNUMBER 1
:SERIALBUS1:PSI5:SEARCH:SETUP:DATA:FR
AMEINSLOT:SNUMBER?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:
DATA:FRAMEINSLOT:SNUMBER 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor?

Function Queries all PSI5 signal error search settings.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor?
```
<x> = 1 to 4

:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:CLOCk

Function Sets or queries the PSI5 signal clock error search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:CLOCk {<Boolean>}
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:CLOCk?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:CLOCK ON
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:CLOCK?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:CLOCK 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FNUMber

Function Sets or queries the PSI5 signal frame number error search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FNUMber {<Boolean>}
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FNUMber?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:FNUMBER ON
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:FNUMBER?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:FNUMBER 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FRAMe

Function Sets or queries the PSI5 signal frame error search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FRAMe {<Boolean>}
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:FRAMe?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:FRAME ON
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:FRAME?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:FRAME 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:PCRC

Function Sets or queries the PSI5 signal parity/CRC error search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:PCRC {<Boolean>}
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:PCRC?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:PCRC ON
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:PCRC?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:PCRC 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBIT

Function Sets or queries the PSI5 signal start bit error search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBIT {<Boolean>}
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBIT?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:SBIT ON
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:SBIT?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:SBIT 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBOundary

Function Sets or queries the PSI5 signal slot boundary error search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBOundary {<Boolean>}
:SERialbus<x>:PSI5:SEARch[:SETup]:ERRor:SBOundary?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:SBOUNDARY ON
:SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:SBOUNDARY?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:ERROR:SBOUNDARY 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:FRAMeinslot?

Function Queries all PSI5 signal slot specification search settings.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:FRAMeinslot?
```
<x> = 1 to 4

:SERialbus<x>:PSI5:SEARch[:SETup]:FRAMeinslot:SNUMber

Function Sets or queries the slot number of the PSI5 signal slot specification search.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:FRAMeinslot:SNUMber {<NRf>}
:SERialbus<x>:PSI5:SEARch[:SETup]:FRAMeinslot:SNUMber?
```
<x> = 1 to 4
<NRf> = 1 to 6

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:FRAMEINSLOT:SNUMBER 1
:SERIALBUS1:PSI5:SEARCH:SETUP:FRAMEINSLOT:SNUMBER?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:FRAMEINSLOT:SNUMBER 1
```

:SERialbus<x>:PSI5:SEARch[:SETup]:MODE

Function Sets or queries the PSI5 signal search mode setting.

Syntax
```
:SERialbus<x>:PSI5:SEARch[:SETup]:MODE {DATA|ERRor|FRAMeinslot|SBIT|SYNC}
:SERialbus<x>:PSI5:SEARch[:SETup]:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:PSI5:SEARCH:SETUP:MODE DATA
:SERIALBUS1:PSI5:SEARCH:SETUP:MODE?
-> :SERIALBUS1:PSI5:SEARCH:SETUP:MODE DATA
```

:SERialbus<x>:PSI5:TRENd<y>?

Function Queries all PSI5 signal analysis trend display settings.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>?
```
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:PSI5:TRENd<y>:ASCale

Function Executes auto scaling of the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:ASCale
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:PSI5:TREND1:ASCALE
```

:SERialbus<x>:PSI5:TRENd<y>:CURSor?

Function Queries all PSI5 signal analysis trend display cursor measurement settings.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:CURSor?
```
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:PSI5:TRENd<y>:CURSor:DT:VALue?

Function Queries the time value between cursors on the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:CURSor:DT:VALue?
```
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:PSI5:TRENd<y>:CURSor:DV:VALue?

Function Queries the vertical value between cursors on the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:CURSor:DV:VALue?
```
<x> = 1 to 4
<y> = 1 to 4

:SERialbus<x>:PSI5:TRENd<y>:CURSor:MODE

Function Sets or queries the cursor mode on the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:CURSor:MODE {<Boolean>}
:SERialbus<x>:PSI5:TRENd<y>:CURSor:MODE?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:PSI5:TREND1:CURSOR:MODE ON
:SERIALBUS1:PSI5:TREND1:CURSOR:MODE?
-> :SERIALBUS1:PSI5:TREND1:CURSOR:MODE 1
```

:SERialbus<x>:PSI5:TRENd<y>:CURSor:POSition<z>

Function Sets or queries the position of the specified cursor on the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:CURSor:POSition<z> {<NRf>}
:SERialbus<x>:PSI5:TRENd<y>:CURSor:POSition<z>?
```
<x> = 1 to 4
<y> = 1 to 4
<z> = 1 or 2
<NRf> = −5 to 5 divisions (in steps of 10 divisions/display record length)

Example
```
:SERIALBUS1:PSI5:TREND1:CURSOR:POSITION1 -5
:SERIALBUS1:PSI5:TREND1:CURSOR:POSITION1?
-> :SERIALBUS1:PSI5:TREND1:CURSOR:POSITION1 -5
```

:SERialbus<x>:PSI5:TRENd<y>:CURSor:T<z>:VALue?

Function Queries the time value at the specified cursor on the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:CURSor:T<z>:VALue?
```
<x> = 1 to 4

<y> = 1 to 4

<z> = 1 or 2

:SERialbus<x>:PSI5:TRENd<y>:CURSor:V<z>:VALue?

Function Queries the vertical value at the specified cursor on the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:CURSor:V<z>:VALue?
```
<x> = 1 to 4

<y> = 1 to 4

<z> = 1 or 2

:SERialbus<x>:PSI5:TRENd<y>:DISPlay

Function Sets or queries the on/off status of the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:DISPlay {<Boolean>}
:SERialbus<x>:PSI5:TRENd<y>:DISPlay?
```
<x> = 1 to 4

<y> = 1 to 4

Example
```
:SERIALBUS1:PSI5:TREND1:DISPLAY ON
:SERIALBUS1:PSI5:TREND1:DISPLAY?
-> :SERIALBUS1:PSI5:TREND1:DISPLAY 1
```

:SERialbus<x>:PSI5:TRENd<y>:HRANge

Function Sets or queries the PSI5 signal analysis trend display source window.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:HRANge {MAIN|Z1|Z2}
:SERialbus<x>:PSI5:TRENd<y>:HRANge?
```
<x> = 1 to 4

<y> = 1 to 4

Example
```
:SERIALBUS1:PSI5:TREND1:HRANGE MAIN
:SERIALBUS1:PSI5:TREND1:HRANGE?
-> :SERIALBUS1:PSI5:TREND1:HRANGE MAIN
```

:SERialbus<x>:PSI5:TRENd<y>:SOURce

Function Sets or queries the target slot number of the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:SOURce {<NRf>}
:SERialbus<x>:PSI5:TRENd<y>:SOURce?
```
<x> = 1 to 4

<y> = 1 to 4

<NRf> = 1 to 6

Example
```
:SERIALBUS1:PSI5:TREND1:SOURCE 1
:SERIALBUS1:PSI5:TREND1:SOURCE?
-> :SERIALBUS1:PSI5:TREND:SOURCE 1
```

:SERialbus<x>:PSI5:TRENd<y>:VERTical

Function Sets or queries the vertical range of the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:VERTical {<NRf>,<NRf>}
:SERialbus<x>:PSI5:TRENd<y>:VERTical?
```
<NRf> = 1.0000E+31 to 1.0000E+31

<x> = 1 to 4

<y> = 1 to 4

Example
```
:SERIALBUS1:PSI5:TREND1:VERTICAL 1,-1
:SERIALBUS1:PSI5:TREND1:VERTICAL?
-> :SERIALBUS1:PSI5:TREND1:VERTICAL 1.0000000E+00,-1.0000000E+00
```

:SERialbus<x>:PSI5:TRENd<y>:VTDisplay

Function Sets or queries the on/off status of the VT waveform display on the PSI5 signal analysis trend display.

Syntax
```
:SERialbus<x>:PSI5:TRENd<y>:VTDisplay {<Boolean>}
:SERialbus<x>:PSI5:TRENd<y>:VTDisplay?
```
<x> = 1 to 4

<y> = 1 to 4

Example
```
:SERIALBUS1:PSI5:TREND1:VTDISPLAY ON
:SERIALBUS1:PSI5:TREND1:VTDISPLAY?
-> :SERIALBUS1:PSI5:TREND1:VTDISPLAY 1
```

:SERialbus<x>:RWINdow

Function Sets or queries in which zoom window, ZOOM1 or ZOOM2, the detected section will be displayed.

Syntax
```
:SERialbus<x>:RWINdow {Z1|Z2}
:SERialbus<x>:RWINdow?
```
<x> = 1 to 4 (1 only for 2-channel models)

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 the specified 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:FA
ST:USETup:DATA<y>:SIZE {<NRf>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FA
ST:USETup:DATA<y>:SIZE?
```
<x> = 1 to 4
<y> = 1 to 4
<NRf> = 0 to 24

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:U
SETUP:DATA1:SIZE 24
:SERIALBUS1:SENT:ANALYZE:SETUP:FAST:U
SETUP:DATA1:SIZE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FAS
T:USETUP:DATA1:SIZE 24
```

:SERialbus<x>:SENT[:ANALyze]:SETup:FORMat?

Function Queries all SENT signal analysis format settings.

Syntax
```
:SERialbus<x>:SENT[:ANALyze]:SETup:FO
RMat?
```
<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:FO
RMat: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:FO
RMat: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:FO
RMat:SAComm:BIT<y>?
```
<x> = 1 to 4
<y> = 0, 1

: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|PO
PTion}
: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:FOR
MAT: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:FO
RMat:CRCType {LEGacy|RECommended}
:SERialbus<x>:SENT[:ANALyze]:SETup:FO
RMat:CRCType?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT
:CRCTYPE LEGACY
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT
:CRCTYPE?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FOR
MAT: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:FO
RMat:CTICk {<NRf>}
:SERialbus<x>:SENT[:ANALyze]:SETup:FO
RMat:CTICk?
```
<x> = 1 to 4
<NRf> = 1.00 μs to 100.00 μs (resolution:0.01 μs)

Example
```
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT
:CTICK 0.000001
:SERIALBUS1:SENT:ANALYZE:SETUP:FORMAT
:CTICK?
-> :SERIALBUS1:SENT:ANALYZE:SETUP:FOR
MAT: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 {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 or 2

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` (number of bytes, 8 digits)(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 DLM2000 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 DLM2000 returns “NAN.”

:SERialbus<x>:SENT:SEARch:SETup?

Function Queries all SENT signal search condition 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: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]:ERRor:SCPulses

Function Sets or queries the SENT signal search consecutive calibration pulse error.

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]: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 user data of the SENT signal search fast channel.

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 {BETWeen|EQUal|GREater|LESS|NOTBetween|NOTEqul}
: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 BETWEEN
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DATA1:CONDITION BETWEEN
```

: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 DLM2000 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]:FDATa:DNIBbles:PFORmat

Function Sets or queries the data input format, which is one of the fast channel nibble data conditions, for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:PFORmat {BINary|HEXa}
:SERialbus<x>:SENT:SEARch[:SETup]:FDATa:DNIBbles:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:PFORMAT BINARY
:SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:PFORMAT?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FDATA:DNIBBLES:PFORMAT BINARY
```

: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]:FSAComm:PFORmat

Function Sets or queries the data input format, which is one of the fast channel status and communication nibble data conditions, for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:FSA
Comm:PFORmat {BINary|HEXa}
:SERialbus<x>:SENT:SEARch[:SETup]:FSA
Comm:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:FSACOMM
:PFORMAT BINARY
:SERIALBUS1:SENT:SEARCH:SETUP:FSACOMM
:PFORMAT?
-> :SERIALBUS1:SENT:SEARCH:SETUP:FSAC
OMM:PFORMAT BINARY
```

:SERialbus<x>:SENT:SEARch[:SETup]:MODE

Function Sets or queries the SENT signal search mode.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:MO
DE {EFASt|ERRor|ESLow|FDATa|FSAComm|S
DATa}
:SERialbus<x>:SENT:SEARch[:SETup]:MO
DE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:MO
DE ERRor
:SERIALBUS1:SENT:SEARCH:SETUP:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:MO
DE ERRor
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa?

Function Queries all slow channel data settings for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta?
```
<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]:SDA
Ta: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]:SDA
Ta:ENHanced:CBIT {D12Bit|D16Bit}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:CBIT?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:CBIT D12Bit
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:CBIT?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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]:SDA
Ta: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]:SDA
Ta: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]:SDA
Ta:ENHanced:D12Bit:DATA:CONDition {BE
TWeen|EQUal|FALSe|GREater|LESS|NOTBet
ween|NOTEqul|TRUE}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D12Bit:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:CONDITION FALSE
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:S
DATA:ENHANCED:D12BIT:DATA:CONDITI
ON 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 DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:DECIMAL1 123
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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:HE
Xa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED: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:MO
DE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D12Bit:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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:PATTe
rn {<String>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D12Bit:DATA:PATTern?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:S
DATA:ENHANCED:D12BIT:DATA:PATTE
RN "111101010101"
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:PATTERN?
-> :SERIALBUS1:SENT:SEARCH:SETUP:
SDATA:ENHANCED:D12BIT:DATA:PATTE
RN "111101010101"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D12Bit:DATA:PFORmat

Function Sets or queries the data input format, which is one of the slow channel enhanced type 12-bit data and 8-bit ID data conditions, for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]
:SDATa:ENHanced:D12Bit:DATA:PFORm
at {BINary|HEXa}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D12Bit:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:PFORMAT BINARY
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:DATA:PFORMAT?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A:ENHANCED:D12BIT:DATA:PFORMAT BINARY
```

: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]:SDA
Ta: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]:SDA
Ta:ENHanced:D12Bit:ID:CONDition {BETW
een|EQUal|GREater|LESS|NOTBetween|NOT
Equl}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D12Bit:ID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:ID:CONDITION BETWEEN
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:ID:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:
SDATA:ENHANCED:D12BIT:ID:CONDITI
ON BETWEEN
```

: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 binary notation.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]
:SDATa:ENHanced:D12Bit:ID:DECimal
<y> {<NRf>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D12Bit:ID:HEXa<y>?
```
<x> = 1 to 4
<y> = 1 or 2

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:ID:DECIMAL1 0
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:ID:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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[:SETu
p]:SDATa:ENHanced:D12Bit:ID:MO
DE {<Boolean>}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D12Bit:ID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:ID:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D12BIT:ID:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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]:SDA
Ta: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]:SDA
Ta: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]:SDA
Ta:ENHanced:D16Bit:DATA:CONDition {BE
TWeen|EQUal|FALSe|GREater|LESS|NOTBet
ween|NOTEqul|TRUE}
:SERialbus<x>:SENT:SEARch[:SETup]:SDA
Ta:ENHanced:D16Bit:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:CONDITION FALSE
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:S
DATA:ENHANCED:D16BIT:DATA:CONDITI
ON 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 DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:DECIMAL1 123
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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:E
NHANCED: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]:SDA
Ta:ENHanced:D16Bit:DATA:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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]:SDA
Ta: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:E
NHANCED:D16BIT:DATA:PATTERN?
-> :SERIALBUS1:SENT:SEARCH:SETUP:
SDATA:ENHANCED:D16BIT:DATA:PATTE
RN "1111000011110000"
```

:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:ENHanced:D16Bit:DATA:PFORmat

Function Sets or queries the data input format, which is one of the slow channel enhanced type 16-bit data and 4-bit ID data conditions, for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:
SDATa:ENHanced:D16Bit:DATA:PFORm
at {BINary|HEXa}
:SERialbus<x>:SENT:SEARch[:SETup]:
SDATa:ENHanced:D16Bit:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:PFORMAT BINARY
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:DATA:PFORMAT?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A:ENHANCED:D16BIT:DATA:PFORMAT BINARY
```

: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]:SDA
Ta: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]:
SDATa:ENHanced:D16Bit:ID:CONDiti
on {BETWeen|EQUal|GREater|LESS|NOTBet
ween|NOTEqul}
:SERialbus<x>:SENT:SEARch[:SETup]:
SDATa:ENHanced:D16Bit:ID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:ID:CONDITION BETWEEN
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:ID:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:
SDATA:ENHANCED:D16BIT:ID:CONDITI
ON BETWEEN
```

: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]:
SDATa:ENHanced:D16Bit:ID:DECimal<y>?
```
<x> = 1 to 4
<y> = 1 or 2

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:ID:DECIMAL1 0
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:ID:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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[:SETup]:
SDATa:ENHanced:D16Bit:ID:MODE {<Boole
an>}
:SERialbus<x>:SENT:SEARch[:SETup]:
SDATa:ENHanced:D16Bit:ID:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:ID:MODE ON
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:E
NHANCED:D16BIT:ID:MODE?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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]:SDA
Ta: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]:SDA
Ta: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]:
SDATa:SHORt:DATA:CONDition {BETWeen|E
QUal|FALSe|GREater|LESS|NOTBetween|NO
TEqul|TRUE}
:SERialbus<x>:SENT:SEARch[:SETup]:
SDATa:SHORt:DATA:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:S
HORT:DATA:CONDITION FALSE
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:S
HORT:DATA:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A:SHORT:DATA:CONDITION FALSE
```

: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]:SDA
Ta:SHORt:DATA:DECimal<y> {<NRf>}
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:S
HORt:DATA:DECIMAL1 123
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:S
HORt:DATA:DECIMAL1?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDAT
A: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]:SDA
Ta:SHORt:DATA:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:S
HORT: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:DATA:PFORmat

Function Sets or queries the data input format, which is one of the slow channel short type data conditions, for SENT signal searching.

Syntax
```
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:PFORmat {BINary|HEXa}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:DATA:PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:PFORMAT BINARY
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:PFORMAT?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:DATA:PFORMAT BINARY
```

: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 {BETWeen|EQUal|GREater|LESS|NOTBetween|NOTEqul}
:SERialbus<x>:SENT:SEARch[:SETup]:SDATa:SHORt:ID:CONDition?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:CONDITION BETWEEN
:SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:CONDITION?
-> :SERIALBUS1:SENT:SEARCH:SETUP:SDATA:SHORT:ID:CONDITION BETWEEN
```

: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

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<y> {<NRf>}
:SERialbus<x>:SENT:TRENd<y>:CURSor:POSition<y>?
```
<x> = 1 to 4
<y> = 1 to 4
<z> = 1 or 2
<NRf> = −5 to 5 divisions (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>:HRANge

Function Sets or queries the SENT signal analysis trend display source window.

Syntax
```
:SERialbus<x>:SENT:TRENd<y>:HRANge {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 MAIN
```

: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 {<String>}
: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>:SOURce {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 FAST
```

: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>:VERTical {<NRf>,<NRf>}
:SERialbus<x>:SENT:TRENd<y>:VERTical?
```
<x> = 1 to 4
<y> = 1 to 4
<NRf> = –1.0000E+31 to 1.0000E+31

Example
```
:SERIALBUS1:SENT:TREND1:VERTICAL 1,-1
:SERIALBUS1:SENT:TREND1:VERTICAL?
-> :SERIALBUS1:SENT:TREND1:VERTICAL 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>:VTDisplay {<Boolean>}
:SERialbus<x>:SENT:TRENd<y>:VTDisplay?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:SENT:TREND1:VTDISPLAY ON
:SERIALBUS1:SENT:TREND1:VTDISPLAY?
-> :SERIALBUS1:SENT:TREND1:
VTDISPLAY 1
```

:SERialbus<x>:SOURce?

Function Queries all analysis and search settings.

Syntax
```
:SERialbus<x>:SOURce?
```
<x> = 1 to 4 (1 only for 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 only for 2-channel models)
<y> of CHANnel<y> = 1 to 4 (1 or 2 for 2-channel models)
<y> of MATH<y> = 1 or 2 (MATH not available for 2-channel models)

:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:HYSTeresis

Function Sets or queries a 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 only for 2-channel models)
<y> of CHANnel<y> = 1 to 4 (1 or 2 for 2-channel models)
<y> of MATH<y> = 1 or 2 (MATH not available for 2-channel models)
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:SOURCE:CHANNEL1:HYSTERES
IS 1
:SERIALBUS1:SOURCE:CHANNEL1:HYSTERES
IS?
-> :SERIALBUS1:SOURCE:CHANNEL1:HYSTER
ESIS 1.000E+00
```

:SERialbus<x>:SOURce:{CHANnel<y>|MATH<y>}:LEVel

Function Sets or queries a source waveform level.

Syntax
```
:SERialbus<x>:SOURce:{CHANnel<y>|MATH
<y>}:LEVel {<Voltage>,<Voltage>}:SERi
albus<x>:SOURce:{CHANnel<y>|MATH<y>}:
LEVel?
```
<x> = 1 to 4 (1 only for 2-channel models)
<y> of CHANnel<y> = 1 to 4 (1 or 2 for 2-channel models)
<y> of MATH<y> = 1 or 2 (MATH not available for 2-channel models)
<Voltage> = See the DLM2000 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 only for 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 only for 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup?

Function Queries all user-defined bus signal analysis settings.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SET
up?
```
<x> = 1 to 4 (1 only for 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup:BRATe

Function Sets or queries the bit rate for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:BRATe {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:BRATe?
```
<x> = 1 to 4 (1 only for 2-channel models)
<NRf> = 1000 to 50000000 (bps)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:BR
ATE 1000
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:BR
ATE?
-> :SERIALBUS1:SPATTERN:ANALYZE:
SETUP:BRATE 1000
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk?

Function Queries all clock signal settings for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CLOCk?
```
<x> = 1 to 4 (1 only for 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:MODE

Function Sets or queries the clock signal enable or disable status for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CLOCk:MODE {<Boolean>}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CLOCk:MODE?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CL
OCK:MODE ON
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CL
OCK:MODE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP
:CLOCK:MODE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:POLarity

Function Sets or queries the clock signal slope for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CLOCk:POLarity {FALL|RISE}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CLOCk:POLarity?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CL
OCK:POLARITY FALL
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CL
OCK:POLARITY?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP
:CLOCK:POLARITY FALL
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CLOCk:SOURce

Function Sets or queries the clock signal for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CLOCk:SOURce {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CLOCk:SOURce?
```
<x> = 1 to 4 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CL
OCK:SOURCE 1
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:CL
OCK:SOURCE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP
:CLOCK:SOURCE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS?

Function Queries all chip select signal settings for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CS?
```
<x> = 1 to 4 (1 only for 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup:CS:ACTive

Function Sets or queries the chip select signal active state for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CS:ACTive {HIGH|LOW}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CS:ACTive?
```
<x> = 1 to 4 (1 only for 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 chip select signal for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CS:SOURce {<NRf>|NONE}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:CS:SOURce?
```
<x> = 1 to 4 (1 only for 2-channel models)
<NRf> = 1 to 4 (NRf not available for 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 data signal settings for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:DATA?
```
<x> = 1 to 4 (1 only for 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:ACTive

Function Sets or queries the data signal active state for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:DATA:ACTive {HIGH|LOW}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:DATA:ACTive?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DA
TA:ACTIVE HIGH
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DA
TA:ACTIVE?
-> :SERIALBUS1:SPATTERN:ANALYZE:SETUP
:DATA:ACTIVE HIGH
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:DATA:SOURce

Function Sets or queries the data signal for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:DATA:SOURce {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:DATA:SOURce?
```
<x> = 1 to 4 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DA
TA:SOURCE 1
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:DA
TA:SOURCE?
-> :SERIALBUS1:SPATTERN:ANALYZE:
SETUP:DATA:SOURCE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh?

Function Queries all latch signal settings for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:LATCh?
```
<x> = 1 to 4 (1 only for 2-channel models)

:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:POLarity

Function Sets or queries the latch signal slope for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:LATCh:POLarity {FALL|RISE}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:LATCh:POLarity?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LA
TCH:POLARITY FALL
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LA
TCH:POLARITY?
-> :SERIALBUS1:SPATTERN:ANALYZE:
SETUP:LATCH:POLARITY FALL
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:LATCh:SOURce

Function Sets or queries the latch signal for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:LATCh:SOURce {<NRf>|NONE}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:LATCh:SOURce?
```
<x> = 1 to 4 (1 only for 2-channel models)
<NRf> = 1 to 4 (NRf not available for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LA
TCH:SOURCE 1
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:LA
TCH:SOURCE?
-> :SERIALBUS1:SPATTERN:ANALYZE:
SETUP:LATCH:SOURCE 1
```

:SERialbus<x>:SPATtern[:ANALyze]:SETup:SPOint

Function Sets or queries the analysis start point for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:SPOint {<NRf>}
:SERialbus<x>:SPATtern[:ANALyze]:SETu
p:SPOint?
```
<x> = 1 to 4 (1 only for 2-channel models)
<NRf> = −5 to 5 (divisions)

Example
```
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:SP
OINT -5
:SERIALBUS1:SPATTERN:ANALYZE:SETUP:SP
OINT?
-> :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 only for 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 only for 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 only for 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 only for 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 DLM2000 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 only for 2-channel models)

Example
```
:SERIALBUS1:SPATtern:SEARCH:SELECT? MAXIMUM
-> :SERIALBUS1:SPATtern:SEARCH:SELECT 100
```

Description If there are no detected points, the DLM2000 returns “NAN.”

:SERialbus<x>:SPATtern:SEARch:SETup?

Function Queries all user-defined bus signal search condition settings.

Syntax
```
:SERialbus<x>:SPATtern:SEARch:SETup?
```
<x> = 1 to 4 (1 only for 2-channel models)

:SERialbus<x>:SPATtern:SEARch:SETup:BITSize

Function Sets or queries the bit length setting for user-defined bus signal analysis.

Syntax
```
:SERialbus<x>:SPATtern:SEARch:SETup:BITSize {<NRf>}
:SERialbus<x>:SPATtern:SEARch:SETup:BITSize?
```
<x> = 1 to 4 (1 only for 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 condition for user-defined bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:SPATtern:SEARch:SETup:HEXa {<String>}
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:SEARCH:SETUP:HEXA "12"
```

:SERialbus<x>:SPATtern:SEARch:SETup:PATTern

Function Sets or queries the data condition for user-defined bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:SPATtern:SEARch:SETup:PATTern {<String>}
:SERialbus<x>:SPATtern:SEARch:SETup:PATTern?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:SEARCH:SETUP:PATTERN "00110101"
:SERIALBUS1:SPATTERN:SEARCH:SETUP:PATTERN?
-> :SERIALBUS1:SPATTERN:SEARCH:SETUP:PATTERN "00110101"
```

:SERialbus<x>:SPATtern:SEARch:SETup:PFORmat

Function Sets or queries the input format, which is one of the data conditions, for user-defined bus signal searching.

Syntax
```
:SERialbus<x>:SPATtern:SEARch:SETup:PFORmat {BINary|HEXa}
:SERialbus<x>:SPATtern:SEARch:SETup:PFORmat?
```
<x> = 1 to 4 (1 only for 2-channel models)

Example
```
:SERIALBUS1:SPATTERN:SEARCH:SETUP:PFORMAT BINARY
:SERIALBUS1:SPATTERN:SEARCH:SETUP:PFORMAT?
-> :SERIALBUS1:SPATTERN:SEARCH:SETUP:PFORMAT BINARY
```

: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 clock signal settings for SPI bus signal analysis.

Syntax `:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk?`
<x> = 1 to 4

:SERialbus<x>:SPI[:ANALyze]:SETup:CLOCk:POLarity

Function Sets or queries the clock signal slope for SPI bus signal analysis.

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 clock signal for SPI bus signal analysis.

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 or 2

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 chip select signal settings for SPI bus signal analysis.

Syntax `:SERialbus<x>:SPI[:ANALyze]:SETup:CS?`
<x> = 1 to 4

:SERialbus<x>:SPI[:ANALyze]:SETup:CS:ACTive

Function Sets or queries the chip select signal active state for SPI bus signal analysis.

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 chip select signal for SPI bus signal analysis.

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 or 2

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 data signal settings for SPI bus signal analysis.

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 data signal for SPI bus signal analysis.

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 or 2

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 data field size for SPI bus signal analysis.

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 on/off status of grouping for SPI bus signal analysis when there is no chip select signal.

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 SPI bus signal analysis when there is no chip select signal.

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 data signal wiring system (three-wire or four-wire) for SPI bus signal analysis.

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 data MSB and LSB bits for SPI bus signal analysis.

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 of 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 DLM2000 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 a 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 DLM2000 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:SELe
ct? {MAXimum}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:SELECT? MAXIMU
M
-> :SERIALBUS1:SPI:SEARCH:SELECT 100
```

Description If there are no detected points, the DLM2000 returns “NAN.”

:SERialbus<x>:SPI:SEARch:SETup?

Function Queries all SPI bus signal search condition settings.

Syntax
```
:SERialbus<x>:SPI:SEARch:SETup?
```
<x> = 1 to 4

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>?

Function Queries all SPI bus signal search data settings.

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 starting position of data comparison 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:BC
OUNT 0
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:BC
OUNT?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1
:BCOUNT 0
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:CONDition

Function Sets or queries the data comparison condition (true or false) 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:CO
NDITION DONTCARE
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:CO
NDITION?
-> :SERIALBUS1:SPI:SEARCH:SETUP:
DATA1:CONDITION DONTCARE
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:DBYTe

Function Sets or queries the data size (in bytes) 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:DB
YTE 1
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:DB
YTE?
-> :SERIALBUS1:SPI:SEARCH:SETUP:
DATA1:DBYTE 1
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:HEXa<z>

Function Sets the data for SPI bus signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA
<y>:HEXa<z> {<String>}
```
<x> = 1 to 4
<y> = 1 or 2
<z> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:HE
XA1 "AB"
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:MODE

Function Sets or queries the data enable/disable condition 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>:SPI:SEARch[:SETup]:DATA<y>:PATTern<z>

Function Sets or queries the data value for SPI bus signal searching in binary notation.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:PATTern<z> {<String>}
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:PATTern<z>?
```
<x> = 1 to 4

<y> = 1 or 2

<z> = 1 to 4

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:PATTERN1 "11001010"
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:PATTERN1?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1:PATTERN1 "11001010"
```

:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:PFORmat

Function Sets or queries the input format, which is one of the data conditions, for SPI bus signal searching.

Syntax
```
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:PFORmat {BINary|HEXa}
:SERialbus<x>:SPI:SEARch[:SETup]:DATA<y>:PFORmat?
```
<x> = 1 to 4

<y> = 1 or 2

Example
```
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:PFORMAT BINARY
:SERIALBUS1:SPI:SEARCH:SETUP:DATA1:PFORMAT?
-> :SERIALBUS1:SPI:SEARCH:SETUP:DATA1:PFORMAT BINARY
```

:SERialbus<x>:TYPE

Function Sets or queries search type.

Syntax
```
:SERialbus<x>:TYPE {CANFDbus|CANBus|CXPIbus|FLEXray|I2CBus|LINBus|PSI5|SENT|SPATtern|SPIBus|UART}
:SERialbus<x>:TYPE?
```
<x> = 1 to 4 (1 only for 2-channel models)

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 UART signal analysis bus settings.

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 DLM2000 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 byte space setting for UART signal analysis.

Syntax
```
:SERialbus<x>:UART[:ANALyze]:SETup:BSPace {<Time>}
:SERialbus<x>:UART[:ANALyze]:SETup:BSPace?
```
<x> = 1 to 4
<Time> = See the DLM2000 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 the on/off status of grouping for UART signal analysis.

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 polarity setting for UART signal analysis.

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 source signal for UART signal analysis.

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 or 2

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 {FU
LL|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 FU
LL
```

: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(ASC
II),Information"
```

:SERialbus<x>:UART:DETail:LIST:VALue?

Function Queries all of the data for the specified analysis number in the UART signal analysis result list.

Syntax
```
:SERialbus<x>:UART:DETail:LIST:VAL
ue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 to 4

<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:SERIALBUS1:UART:DETAIL:LIST:VALUE? 0
-> :SERIALBUS1:UART:DETAIL:LIST:VAL
UE "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:SELe
ct {<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 DLM2000 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:SELe
ct? {MAXimum}
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SELECT? MAXIM
UM
-> :SERIALBUS1:UART:SEARCH:SELECT 100
```

Description If there are no detected points, the DLM2000 returns “NAN.”

:SERialbus<x>:UART:SEARch:SETup?

Function Queries all UART signal search condition settings.

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 bus 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:AS
CII "TEST"
```

:SERialbus<x>:UART:SEARch:SETup:DATA:CONDition

Function Sets or queries the data comparison condition (true or 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:CO
NDITION DONTCARE
:SERIALBUS1:UART:SEARCH:SETUP:DATA:CO
NDITION?
-> :SERIALBUS1:UART:SEARCH:SETUP:
DATA: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 bus 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:CS
ENSITIVE ON
:SERIALBUS1:UART:SEARCH:SETUP:DATA:CS
ENSITIVE?
-> :SERIALBUS1:UART:SEARCH:SETUP:
DATA: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 bytes

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:DB
YTE 1
:SERIALBUS1:UART:SEARCH:SETUP:DATA:DB
YTE?
-> :SERIALBUS1:UART:SEARCH:SETUP:
DATA:DBYTE 1
```

:SERialbus<x>:UART:SEARch:SETup:DATA:HEXa<y>

Function Sets the data for UART signal searching in hexadecimal notation.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA:
HEXa<y> {<String>}
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:HE
XA1 "12"
```

:SERialbus<x>:UART:SEARch:SETup:DATA:PATTern<y>

Function Sets or queries the data value for UART signal searching in binary notation.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:DATA:
PATTern<y> {<String>}
:SERialbus<x>:UART:SEARch:SETup:DATA:
PATTern<y>?
```
<x> = 1 to 4
<y> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:PA
TTERN1 "00110101"
:SERIALBUS1:UART:SEARCH:SETUP:DATA:PA
TTERN1?
-> :SERIALBUS1:UART:SEARCH:SETUP:
DATA:PATTERN1 "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|BINary|HEXa}
:SERialbus<x>:UART:SEARch:SETup:DATA:
PFORmat?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:DATA:PF
ORMAT BINARY
:SERIALBUS1:UART:SEARCH:SETUP:DATA:PF
ORMAT?
-> :SERIALBUS1:UART:SEARCH:SETUP:
DATA:PFORMAT BINARY
```

:SERialbus<x>:UART:SEARch:SETup:ERRor?

Function Queries all UART signal search error settings.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:ERR
or?
```
<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:F
RAMING ON
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:F
RAMING?
-> :SERIALBUS1:UART:SEARCH:SETUP:ERRO
R: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:P
ARITY ON
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:P
ARITY?
-> :SERIALBUS1:UART:SEARCH:SETUP:ERRO
R: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:P
MODE EVEN
:SERIALBUS1:UART:SEARCH:SETUP:ERROR:P
MODE?
-> :SERIALBUS1:UART:SEARCH:SETUP:ERRO
R:PMODE EVEN
```

:SERialbus<x>:UART:SEARch:SETup:FORMat

Function Sets or queries the format setting for UART signal analysis.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:FORM
at {BIT7Parity|BIT8Parity|
BIT8Noparity}
:SERialbus<x>:UART:SEARch:SETup:FORM
at?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:FORM
AT BIT7PARITY
:SERIALBUS1:UART:SEARCH:SETUP:FORMAT?
-> :SERIALBUS1:UART:SEARCH:SETUP:FORM
AT BIT7PARITY
```

:SERialbus<x>:UART:SEARch:SETup:MODE

Function Sets or queries the UART signal search mode.

Syntax
```
:SERialbus<x>:UART:SEARch:SETup:MO
DE {DATA|EDATA|ERRor}
:SERialbus<x>:UART:SEARch:SETup:MODE?
```
<x> = 1 to 4

Example
```
:SERIALBUS1:UART:SEARCH:SETUP:MO
DE DATA
:SERIALBUS1:UART:SEARCH:SETUP:MODE?
-> :SERIALBUS1:UART:SEARCH:SETUP:MO
DE DATA
```

:SERialbus<x>:ZLINkage

Function Sets or queries whether or not the analysis numbers of serial-bus signal-analysis results are linked to zoom locations.

Syntax
```
:SERialbus<x>:ZLINkage {<Boolean>}
:SERialbus<x>:ZLINkage?
```
<x> = 1 to 4 (1 only for 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, and starts waveform acquisition. Returns zero if data acquisition is stopped before the specified time period. Returns 1 if waveform acquisition is not stopped within the specified time period.

Syntax `:SSTart? {<NRf>}`

<NRf> = 1 to 36000 (in 100-ms steps. Start and wait.)

<NRf> = 0 (start without any waiting)

<NRf> = −36000 to −1 (in 100-ms steps. Wait without starting.)

Example
```
:SSTART?
-> :SSTART 1
```

Description
- If you specify a positive time value, the DLM2000 starts waveform acquisition in Single Trigger mode and waits until the waveform acquisition is stopped within the specified time period.
- If you set the time value to zero, the DLM2000 starts waveform acquisition and returns zero without waiting until the waveform acquisition is stopped.
- If you specify a negative time value, the DLM2000 does not start waveform acquisition and simply waits until the waveform acquisition in progress is stopped within the specified time period.

5.27 STARt Group

:STARt

Function Starts waveform acquisition.

Syntax `:STARt`

Example `:START`

Description Stop waveform acquisition by using the STOP command.

5.28 STATus Group

The commands in the STATus group are used to make settings and inquiries 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 of the settings for 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 clear 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 information (top of the error queue).

Syntax:
```
:STATus:ERRor?
```

Example:
```
:STATUS:ERROR?
-> 113,"Undefined header"
```

:STATus:FILTer<x>

Function: Sets or queries the transition filter.

Syntax:
```
:STATus:FILTer<x> {BOTH|FALL|NEVer|RI
SE}
:STATus:FILTer<x>?
```
<x> = 1 to 16

Example:
```
:STATUS:FILTER2 RISE
:STATUS:FILTER2?
-> :STATUS:FILTER1 RISE
```

Description: For information about the transition filter, see chapter 6, “Status Reports.”

:STATus:QENable

Function: Sets or queries whether or not messages other than errors will be stored to the error queue (on/off).

Syntax:
```
:STATus:QENable {<Boolean>}
:STATus:QENable?
```

Example:
```
:STATUS:QENABLE ON
:STATUS:QENABLE?
-> :STATUS:QENABLE 1
```

:STATus:QMESsage

Function: Sets or queries whether or not message information will be attached to the response to the STATus:ERRor? query (on/off).

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 waveform acquisition by using the `STARt` command.

5.30 STORe Group

:STORe?

Function Queries all of 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 eight 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 "2008/09/30 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 data-write protection on/off status for the setup data that is 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 the click sound on/off status.

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 DLM2000 Features Guide for this information.

Example
```
:SYSTEM:CLOCK:DATE "2008/09/30"
:SYSTEM:CLOCK:DATE?
-> :SYSTEM:CLOCK:DATE "2008/09/30"
```

: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 the on/off status of the date/time display.

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 date/time setting retrieved using SNTP.

Syntax `:SYSTem:CLOCk:SNTP?`

:SYSTem:CLOCk:SNTP:EXECute

Function Sets the date and time 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 DLM2000 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 DLM2000 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 or not the specified delay value will be applied to time measurement (on/off).

Syntax
```
:SYSTem:DCANcel {<Boolean>}
:SYSTem:DCANcel?
```

Example
```
:SYSTEM:DCANCEL ON
:SYSTEM:DCANCEL?
-> :SYSTEM:DCANCEL 1
```

:SYSTem:FSIZe? (Font Size)

Function Queries all font size settings.

Syntax `:SYSTem:FSIZe?`

:SYSTem:FSIZe:MEASure

Function Sets or queries the font size that is used to display the automatically measured waveform parameters and the cursor measurement values.

Syntax
```
:SYSTem:FSIZe:MEASure {LARGe|SMALl}
:SYSTem:FSIZe:MEASure?
```

Example
```
:SYSTEM:FSIZE:MEASURE LARGE
:SYSTEM:FSIZE:MEASURE?
-> :SYSTEM:FSIZE:MEASURE LARGE
```

: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 of the settings for the feature that automatically turns off the LCD backlight.

Syntax
```
:SYSTem:LCD:AUTO?
```

Example
```
:SYSTEM:LCD:AUTO?
-> :SYSTEM:LCD:AUTO
```

:SYSTem:LCD:AUTO:MODE

Function Sets or queries the on/off status of the feature that automatically turns off the LCD backlight.

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 amount of time until the LCD backlight is turned off.

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 on/off status of the LCD backlight.

Syntax
```
:SYSTem:LCD:MODE {<Boolean>}
:SYSTem:LCD:MODE?
```

Example
```
:SYSTEM:LCD:MODE ON
:SYSTEM:LCD:MODE?
-> :SYSTEM:LCD:MODE 1
```

:SYSTem:LMODe (Legacy Mode)

Function Sets or queries whether to initialize the settings to the default values of legacy models when Default Setup is executed.

Syntax
```
:SYSTem:LMODe {<Boolean>}
:SYSTem:LMODe?
```

Example
```
:SYSTEM:LMODE ON
:SYSTEM:LMODE?
-> :SYSTEM:LMODE 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:OCANcel (Offset Cancel)

Function Sets or queries whether or not the specified offset voltage will be applied to measurement and computation (on/off).

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> = 500 ps to 500 s

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 menu items that appear on the screen.

Trigger Type	Communication Command	Setup Menu	
		Trigger Menu	Trigger Type
Edge trigger	SIMPle	EDGE	—
Trigger on the OR of multiple edge triggers	OR	ENHANCED	Edge OR
Edge qualified trigger	QUALify		Edge Qualify
State trigger	PATTern		State
pulse width trigger	PULSe		Pulse Width
State width trigger	WIDTh		State Width
Serial trigger (CAN, CAN FD, LIN, SENT, PSI5 Airbag, UART, I2C, SPI, FlexRay, and User Define)	CAN*		CAN
	CANFD*		CAN FD
	LIN*		LIN
	SENT*		SENT
	PSI5*		PSI5 Airbag
	UART*		UART
	I2C*		I2C
	SPI*		SPI
	FLEXray*		FlexRay
	SPATtern		User Define
TV trigger (NTSC, PAL, SDTV, HDTV, and User Define)	TV:NTSC		NTSC
	TV:PAL		PAL
	TV:SDTV		SDTV
	TV:HDTV		HDTV
	TV:USERdefine		User Define

* CAN, CANFD, LIN, SENT, PSI5, UART, I2C, SPI, and FLEXray are serial bus options available only for 4-channel models. Communication commands for these serial buses cannot be used on 2-channel models or 4-channel models without serial bus 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 the A->B(n) trigger.

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-on-trigger action.

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 the whether or not the DLM2000 will sound an alarm as an action when trigger conditions are met (on/off).

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 or not the DLM2000 will print a screen capture as an action when trigger conditions are met (on/off).

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 of the settings for email notification that is sent when trigger conditions are met.

Syntax
```
:TRIGger:ACTion:MAIL?
```

Description You can use this command when the optional Ethernet interface is installed.

:TRIGger:ACTion:MAIL:COUNt

Function Sets or queries the upper limit of email notifications that are sent when trigger conditions are met.

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
```

Description You can use this command when the optional Ethernet interface is installed.

:TRIGger:ACTion:MAIL:MODE

Function Sets or queries whether or not the DLM2000 will send email notification 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
```

Description You can use this command when the optional Ethernet interface is installed.

:TRIGger:ACTion:SAVE Function Sets or queries whether or not the DLM2000 will save waveform data to the storage medium as an action when trigger conditions are met (on/off).

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 the action-on-trigger operation.

Syntax
```
:TRIGger:ACTion:STARt
```

Example
```
:TRIGGER:ACTION:START
```

Description Use `:TRIGger:ACTION:STOP` to stop trigger actions.

:TRIGger:ACTion:STOP

Function Stops the action-on-trigger operation.

Syntax
```
:TRIGger:ACTion:STOP
```

Example
```
:TRIGGER:ACTION:STOP
```

Description This command only stops trigger actions and does not cancel the setting. Use `:TRIGger:ACTION:STARt` to resume trigger actions.

: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 the A Delay B trigger.

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 DLM2000 Feature 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:E
FRame:CRC {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:E
FRame:CRC?
```

Example
```
:TRIGGER:ATRIGGER:CAN:EFRAME:CRC ON
:TRIGGER:ATRIGGER:CAN:EFRAME:CRC?
-> :TRIGGER:ATRIGGER:CAN:EFRAME:CRC 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame[:MODE]

Function Sets or queries the CAN bus signal trigger error frame setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:E
FRame[:MODE] {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:E
FRame[:MODE]?
```

Example
```
:TRIGGER:ATRIGGER:CAN:EFRAME:MODE ON
:TRIGGER:ATRIGGER:CAN:EFRAME:MODE?
-> :TRIGGER:ATRIGGER:CAN:EFRAME:
MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:EFRame:STUFF

Function Sets or queries the CAN bus signal trigger stuff error setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:E
FRame:STUFF {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:E
FRame:STUFF?
```

Example
```
:TRIGGER:ATRIGGER:CAN:EFRAME:STUFF ON
:TRIGGER:ATRIGGER:CAN:EFRAME:STUFF?
-> :TRIGGER:ATRIGGER:CAN:EFRAME:
STUFF 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 all ACK settings for 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:M
ODE 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:TY
PE ACK
:TRIGGER:ATRIGGER:CAN:IDDATA:ACK:TY
PE?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:ACK:T
YPE ACK
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA?

Function Queries all data settings for 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 comparison condition, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:
IDData]:DATA:CONDition {BETWeen|EQUal
|FALSe|GREater|LESS|NOTBetween|NOTEqu
l|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:
IDData]:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:CON
DITION BETWEEN
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:CON
DITION?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:
CONDITION BETWEEN
```

: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 DLM2000 Feature Guide for this information.

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DEC
IMAL1 1
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:DEC
IMAL1?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:
DECIMAL1 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:DLC

Function Sets or queries the data length code (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:D
LC 0
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:D
LC?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:
DLC 0
```

: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:END
IAN BIG
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:END
IAN?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:
ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:HEXa<x>

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:HEXa<x>> {<String>}
```
<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:HE
XA1 "12"
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:MODE

Function Sets or queries the data enable/disable condition, 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:MO
DE ON
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MO
DE?
-> :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:MSB
LSB 1,0
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:MSB
LSB?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:
MSBLSB 1,0
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PATTern<x>

Function Sets or queries the data value, which is one of the ID and data conditions, for CAN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PATTern<x> {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PATTern<x>?
```
<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:PATTERN1 "00110101"
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:PATTERN1?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:PATTERN1 "00110101"
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PFORmat

Function Sets or queries the data input format, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:DATA:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:PFORMAT BINARY
:TRIGGER:ATRIGGER:CAN:IDDATA:DATA:PFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:DATA:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[: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 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 all identifier settings for 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 all ID settings for 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 the ID value, 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"
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID and data conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFIER:ID:MODE ON
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFIER:ID:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFIER:ID:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:PATTern

Function Sets or queries the ID pattern, 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:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:IDENtifier:ID:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFIER:ID:PATTERN "11100001111"
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFIER:ID:PATTERN?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFIER:ID:PATTERN "11100001111"
```

: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:MFORmat {EXTended|
STANdard}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:
IDData]:IDENtifier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFI
ER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFI
ER:MFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDENT
IFIER: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:PFORmat {BINary|HE
Xa|MESSage}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:
IDData]:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFI
ER:PFORMAT BINARY
:TRIGGER:ATRIGGER:CAN:IDDATA:IDENTIFI
ER:PFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:IDENT
IFIER:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal?

Function Queries all message and signal settings for 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:
ITEM {<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 message and signal conditions for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:
IDData]:MSIGnal:SELect {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:MSIGN
AL:SELECT MESSAGE
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal?

Function Queries all signal settings for 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 {BET
Ween|EQUal|GREater|LESS|NOTBetween|NO
TEqul}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:
IDData]:MSIGnal:SIGNal:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:
SIGNAL:CONDITION BETWEEN
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:
SIGNAL:CONDITION?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:MSIGN
AL:SIGNAL:CONDITION BETWEEN
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN[:IDData]:MSIGnal:SIGNal:DECimal<x>

Function Sets a signal reference value, which is one of the ID and data conditions, for CAN bus signal triggering in decimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CA
N[:IDData]:MSIGnal:SIGNal:DECimal
<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN[:
IDData]:MSIGnal:SIGNal:DECimal<x>?
```
<x> = 1 or 2
<NRf> = See the DLM2000 Feature Guide for this information.

Example
```
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:
SIGNAL:DECIMAL1 1
:TRIGGER:ATRIGGER:CAN:IDDATA:MSIGNAL:
SIGNAL:DECIMAL1?
-> :TRIGGER:ATRIGGER:CAN:IDDATA:MSIGN
AL: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 IDOR condition settings for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK?

Function Queries all IDOR condition ACK settings for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:ACK?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:MODE

Function Sets or queries the ACK mode, which is one of the IDOR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:ACK:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:ACK:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:MO
DE ON
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDOR:ACK:MO
DE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:ACK:TYPE

Function Sets or queries the ACK condition, which is one of the IDOR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:ACK:TYPE {ACK|ACKBoth|NONack}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:ACK:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:TY
PE ACK
:TRIGGER:ATRIGGER:CAN:IDOR:ACK:TYPE?
-> :TRIGGER:ATRIGGER:CAN:IDOR:ACK:TY
PE ACK
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA?

Function Queries all data settings for the IDOR condition for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:DATA[:MODE]

Function Sets or queries the data enable/disable condition, which is one of the IDOR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:DATA[:MODE] {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:DATA[:MODE]?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:DATA:MO
DE ON
:TRIGGER:ATRIGGER:CAN:IDOR:DATA:MODE?
-> :TRIGGER:ATRIGGER:CAN:IDOR:DATA:MO
DE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier?

Function Queries all identifier settings for the IDOR condition for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>?

Function Queries all IDOR condition ID settings for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR: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 IDOR conditions, for CAN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:IDENtifier:ID<x>:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER
:ID1:HEXA "1AB"
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:MODE

Function Sets or queries the ID enable/disable condition, which is one of the IDOR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:IDENtifier:ID<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR: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:IDENTIF
IER:ID1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:ID<x>:PATTern

Function Sets or queries the ID pattern, which is one of the IDOR conditions, for CAN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
CAN:IDOR:IDENtifier:ID<x>:PATTe
rn {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR: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:IDENTIF
IER:ID1:PATTERN "11100001111"
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MODE

Function Sets or queries the identifier enable/disable condition, which is one of the IDOR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:IDENtifier:MODE {<Boolean>}
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:
IDENTIFIER:MODE ON
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the IDOR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:IDENtifier:MFORmat {EXTended|STAN
dard}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:I
DOR:IDENtifier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:
IDENTIFIER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CAN:IDOR:
IDENTIFIER:MFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDOR:IDENTIF
IER:MFORMAT EXTENDED
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the IDOR conditions, for CAN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:PFORmat {BINary|HEXa|MESSage}
:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:PFORMAT BINARY
:TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:PFORMAT?
-> :TRIGGER:ATRIGGER:CAN:IDOR:IDENTIFIER:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:CAN:IDOR:MSIGnal<x>:MESSage:ITEM

Function Sets the message item, which is one of the IDOR 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 value, which is one of the IDOR 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 setting.

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 source 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}:FLEXray?

Function Queries all FLEXRAY bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BCHannel

Function Sets or queries the channel bus type for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BCHannel {A|B}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:BCHannel?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:BCHANNEL A
:TRIGGER:ATRIGGER:FLEXRAY:BCHANNEL?
-> :TRIGGER:ATRIGGER:FLEXRAY:BCHANNEL 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, or 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
```
: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 all cycle-count settings for 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 {BETWeen|EQUal|GREater|LESS|NOTBetween|NOTEqual}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:CONDITION BETWEEN
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:CCOUNT:CONDITION BETWEEN
```

: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
- For :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition EQUal|GREater|NOTEqul, set the cycle-count setting using :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt1.
- For :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition LESS, set the cycle-count setting using :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt2.
- For :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:CONDition BETWeen|NOTBetween, set the smaller value using :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt1, and set the larger value using :TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:COUNt2.

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:CCOunt:MODE

Function Sets or queries the cycle-count enable/disable condition, 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 all data field settings for 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 position for comparing data field data patterns, 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 {BETWeen|EQUal|FALSe|GREater|LESS|NOTBetween|NOTEqul|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:CONDITION BETWEEN
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:CONDITION BETWEEN
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:DBYTe

Function Sets or queries the number of data field data bytes, 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}:FLEXray[:IDData]:DATA<x>:DECimal<y>?
```
<x> = 1 or 2
<y> = 1 or 2
<NRf> = See the DLM2000 Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:DECIMAL1 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:DECIMAL1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:DECIMAL1 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:ENDian

Function Sets or queries the data field endian setting, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:ENDian {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:ENDian?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:ENDIAN BIG
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:ENDIAN?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:HEXa<y>

Function Sets a data field 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>:
HEXa<y> {<String>}
```
<x> = 1 or 2
<y> = 1 to 8
<String> = Two characters (each representing a byte). Each character can be X or a hexadecimal value from 0 to F.

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:HEXA1 "A9"
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MODE

Function Sets or queries the data field enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:MODE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:MODE?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MSBLsb

Function Sets or queries the MSB and LSB bits for 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>:
MSBLsb {<NRf>,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:MSBLsb?
```
<x> = 1 or 2
<NRf> = See the DLM2000 User's Manual for this information.

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:MSBLSB 7,0
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:MSBLSB?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:MSBLSB 7,0
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:PATTern<y>

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}:FLEXray[:IDData]:DATA<x>:PATTern<y> {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:PATTern<y>?
```
<x> = 1 or 2
<y> = 1 to 8
<String> = Eight characters (each representing a byte). Each character can be 0, 1, or X.

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:PATTERN1 "11011111"
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:PATTERN1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:PATTERN1 "11011111"
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:PFORmat

Function Sets or queries the data field data input format, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:PFORmat?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:PFORMAT BINARY
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:PFORMAT?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:SIGN

Function Sets or queries whether a signed or unsigned data format will be used for the data field (this is one of the ID and data conditions) for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:SIGN {SIGN|UNSign}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:DATA<x>:SIGN?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:SIGN SIGN
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:SIGN?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:DATA1:SIGN SIGN
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID?

Function Queries all frame ID settings for the ID and data conditions for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[: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}:FLEXray[:IDData]:FID:CONDition {BETWeen|EQUal|GREater|LESS|NOTBetween|NOTEqul}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID:CONDITION BETWEEN
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDDATA:FID:CONDITION BETWEEN
```

: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}:FLEXray[:IDData]:FID:ID<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:ID<x>?
```
<x> = 1 or 2
<NRf> = 1 to 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 enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:FID:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[: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 all indicator settings for the ID and data conditions for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:MODE

Function Sets or queries the indicator enable/disable condition, which is one of the ID and data conditions, for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:
MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:MODE?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR:MODE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR: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:NFRame {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:NFRame?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR:NFRAME 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR: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}:FLEXray[:IDData]:INDIcator:PPReamble {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:PPReamble?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR:PPREAMBLE 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR: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:STFRame {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:STFRame?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR:STFRAME 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR: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:SYFRame {0|1|X}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray[:IDData]:INDIcator:SYFRame?
```

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR:SYFRAME 1
:TRIGGER:ATRIGGER:FLEXRAY:IDDATA:INDICATOR: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}:FLEXray:IDOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>?

Function Queries all the settings of the frame IDs, which are OR conditions, for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt?

Function Queries the cycle-count setting for the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).

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 comparison condition for the cycle counts of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:CCOunt:CONDition {BETWe
en|DONTcare|EQUal|GREater|LESS|NOTBet
ween|NOTEqul}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:CCOunt:CONDition?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CC
OUNT:CONDITION BETWEEN
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CC
OUNT:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:
ID1:CCOUNT:CONDITION BETWEEN
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:CCOunt:COUNt<y>

Function Sets or queries the reference value for the cycle counts of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:CCOunt:COUNt<y> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:CCOunt:COUNt<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 0 to 63

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CC
OUNT:COUNT1 100
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:CC
OUNT:COUNT1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:
ID1:CCOUNT:COUNT1 100
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID?

Function Queries all the ID conditions of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:FID?
```
<x> = 1 to 4

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:CONDition

Function Sets or queries the ID comparison condition of the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:FID:CONDition {BETWeen
|EQUal|GREater|LESS|NOTBetween|NOTEq
ul}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:FID:CONDition?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FI
D:CONDITION BETWEEN
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FI
D:CONDITION?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:
ID1:FID:CONDITION BETWEEN
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:FID:ID<y>

Function Sets or queries the ID reference value for the frame IDs for FLEXRAY bus signal triggering (this is one of the OR conditions).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:FID:ID<y> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:FID:ID<y>?
```
<x> = 1 to 4
<y> = 1 or 2
<NRf> = 1 to 2047

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FI
D:ID1 10
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:FI
D:ID1?
-> :TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1
:FID:ID1 10
```

:TRIGger{[:ATRigger]|:BTRigger}:FLEXray:IDOR:ID<x>:MODE

Function Sets or queries the frame ID enable/disable condition, which is one of the OR conditions, for FLEXRAY bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:FLEXr
ay:IDOR:ID<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:MO
DE ON
:TRIGGER:ATRIGGER:FLEXRAY:IDOR:ID1:MO
DE?
-> :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 trigger source for FLEXRAY bus signal triggering.

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 I^{2}C bus signal trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess?

Function Queries all address pattern settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess?

Function Queries all address pattern address settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress?

Function Queries all 10-bit address settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT10ADdress:HEXa

Function Sets the 10-bit address for I^{2}C 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 I^{2}C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT10ADdress:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT10ADdress:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT10ADDRESS:PATTERN "11010111001"
:TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT10ADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT10ADDRESS:PATTERN "11010111001"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]?

Function Queries all 7-bit address settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:HEXa

Function Sets the 7-bit address for I^{2}C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:
HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:BIT7ADDRESS:HEXA "5C"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess[:BIT7ADdress]:PATTern

Function Sets or queries the 7-bit address for I^{2}C 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:
ADDRESS:BIT7ADDRESS:PATTERN "11100110
"
:TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7ADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADDR
ESS:BIT7ADDRESS:PATTERN "11100110"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub?

Function Queries all 7-bit + sub address settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess?

Function Queries all 7-bit address settings for the 7-bit + sub address for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub:ADDRess?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:ADDRess:HEXa

Function Sets the 7-bit address for the 7-bit + sub address for I^{2}C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:
ADDRess:ADDRess:BIT7APsub:ADDRess:HE
Xa {<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 for the 7-bit + sub address for I^{2}C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub:ADDRess:PATT
ern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub:ADDRess:PATT
ern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7APSUB:ADDRESS:PATTE
RN "00111010"
:TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7APSUB:ADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7APSUB:ADDRESS:PATTE
RN "00111010"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress?

Function Queries all sub address settings for the 7-bit + sub address for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub:SADDress?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:BIT7APsub:SADDress:HEXa

Function Sets the sub address for the 7-bit + sub address for I^{2}C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub:SADDress:HE
Xa {<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 for the 7-bit + sub address for I^{2}C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub:SADDress:PAT
Tern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:ADDRess:BIT7APsub:SADDress:PAT
Tern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7APSUB:SADDRESS:PATTE
RN "00111010"
:TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7APSUB:SADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:
ADDRESS:BIT7APSUB:SADDRESS:PATTE
RN "00111010"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:MODE

Function Sets or queries the address condition enable/disable mode for I^{2}C 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:PFORmat

Function Sets or queries the address input format, which is one of the address conditions, for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:PFORMAT BINARY
:TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:PFORMAT?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:ADDRESS:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:ADDRess:TYPE

Function Sets or queries the address type, which is one of the address conditions, for I^{2}C 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 address pattern data settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:BCOunt

Function Sets or queries the position for comparing data patterns for I^{2}C 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 the on/off status of the position for comparing data patterns for I^{2}C 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 comparison condition (true or false) for I^{2}C 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 specified number of data bytes for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:DBYTe?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DB
YTE 1
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:DB
YTE?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:
DATA:DBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:HEXa<x>

Function Sets the data for I^{2}C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:HEXa<x> {<String>}
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:HE
XA1 "AB"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:MODE

Function Sets or queries the data condition enable/disable mode for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:MO
DE ON
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:MO
DE?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:
DATA:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:PATTern<x>

Function Sets or queries the data value, which is one of the data conditions, for I^{2}C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:PATTern<x> {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:PATTern<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:PA
TTERN1 "10001101"
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:PA
TTERN1?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:DATA
:PATTERN1 "10001101"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:ADDRess:DATA:PFORmat

Function Sets or queries the data input format, which is one of the data conditions, for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:A
DDRess:DATA:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:PF
ORMAT BINARY
:TRIGGER:ATRIGGER:I2C:ADDRESS:DATA:PF
ORMAT?
-> :TRIGGER:ATRIGGER:I2C:ADDRESS:
DATA:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall?

Function Queries all general call settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:G
ENeralcall?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress?

Function Queries all general call's 7-bit master address settings for I^{2}C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:G
ENeralcall:BIT7Maddress?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:HEXa

Function Sets the general call's 7-bit master address for I^{2}C bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:
I2C:GENeralcall:BIT7Maddress:HE
Xa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:I2C:GENERALCALL:BIT
7MADDRESS:HEXA "AB"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:BIT7Maddress:PATTern

Function Sets or queries the general call's 7-bit master address for I^{2}C bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:G
ENeralcall:BIT7Maddress:
PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:I2C:G
ENeralcall:BIT7Maddress:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:I2C:GENERALCALL:BIT
7MADDRESS:PATTERN "0010110"
:TRIGGER:ATRIGGER:I2C:GENERALCALL:BIT
7MADDRESS:PATTERN?
-> :TRIGGER:ATRIGGER:I2C:GENERALCALL:
BIT7MADDRESS:PATTERN "0010110"
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:GENeralcall:SBYTe

Function Sets or queries the general call's 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 on/off status of the R/W address bit in 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 trigger type for I²C bus signal triggering.

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 NON ACK ignore mode settings for I²C bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack?
```

:TRIGger{[:ATRigger]|:BTRigger}:I2C:NONack:HSMode

Function Sets or queries whether or not NON ACK will be ignored in high-speed mode for 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 or not NON ACK will be ignored in read access mode for 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 or not NON ACK will be ignored in start bytes for 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 clock signal for I^{2}C bus signal triggering.

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 data signal for I^{2}C bus signal triggering.

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
```

: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 DLM2000 Feature 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:I
DData?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA?

Function Queries all LIN bus signal trigger data settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:CONDition

Function Sets or queries the data conditions for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:CONDition {BETWeen|EQUal|F
ALSe|GREater|LESS|NOTBetween|NOTEqul|
TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:CON
DITION BETWEEN
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:CON
DITION?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:
CONDITION BETWEEN
```

: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:I
DData:DATA:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:DBYTe?
```
<NRf> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DBY
TE 1
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DBY
TE?
-> :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:I
DData:DATA:DECimal<x> {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:DECimal<x>?
```
<x> = 1 or 2

<NRf> = See the DLM2000 Feature Guide for this information.

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DEC
IMAL1 1
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:DEC
IMAL1?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:
DECIMAL1 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:ENDian

Function Sets or queries the data endian setting for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:ENDian {BIG|LITTle}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:ENDian?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:END
IAN BIG
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:END
IAN?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:
ENDIAN BIG
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:HEXa<x>

Function Sets the data for LIN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:HEXa<x> {<String>}
```
<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:HE
XA1 "12"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MODE

Function Sets or queries the data enable/disable condition for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MO
DE ON
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:MO
DE?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:
MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MSBLsb

Function Sets or queries the data MSB and LSB bits for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:MSBLsb?
```
<NRf>,<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:PATTern<x>

Function Sets or queries the data value for LIN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:PATTern<x> {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:PATTern<x>?
```
<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:PATTERN1 "00110101"
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:PATTERN1?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:PATTERN1 "00110101"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:PFORmat

Function Sets or queries the input format, which is one of the data conditions, for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:PFORMAT BINARY
:TRIGGER:ATRIGGER:LIN:IDDATA:DATA:PFORMAT?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:DATA:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:DATA:SIGN

Function Sets or queries whether signed or unsigned data format will be used, which is one of the data conditions, 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>}
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFIER:ID:HEXA "1E"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:ID:MODE

Function Sets or queries the ID enable/disable condition, 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 value 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:IDENTIFI
ER:ID:PATTERN "101100"
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFI
ER:ID:PATTERN?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:IDENT
IFIER:ID:PATTERN "101100"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDData:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID and data conditions, for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:IDENtifier:PFORmat {BINary|HEXa
}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DData:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFI
ER:PFORMAT BINARY
:TRIGGER:ATRIGGER:LIN:IDDATA:IDENTIFI
ER:PFORMAT?
-> :TRIGGER:ATRIGGER:LIN:IDDATA:IDENT
IFIER:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR?

Function Queries all IDOR condition settings for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier?

Function Queries all IDOR condition ID settings for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR:IDENtifier?
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>?

Function Queries an ID value, which is one of the IDOR condition settings, for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR: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 IDOR conditions, for LIN bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR: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 IDOR conditions, for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR:IDENtifier:ID<x>:
MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR: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:IDENTIF
IER:ID1:MODE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:ID<x>:PATTern

Function Sets or queries the ID pattern, which is one of the IDOR conditions, for LIN bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:
IDOR:IDENtifier:ID<x>:PATTern {<Strin
g>}
: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:IDENTIF
IER:ID1:PATTERN "101100"
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:MODE

Function Sets or queries the ID enable/disable condition for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR:IDENtifier:MODE {<Boolean>}
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDOR:
IDENTIFIER:MODE ON
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:IDOR:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the IDOR conditions, for LIN bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR:IDENtifier:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:I
DOR:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:LIN:IDOR:
IDENTIFIER:PFORMAT BINARY
:TRIGGER:ATRIGGER:LIN:IDOR:
IDENTIFIER:PFORMAT?
-> :TRIGGER:ATRIGGER:LIN:IDOR:IDENTIF
IER:PFORMAT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:LIN:MODE

Function Sets or queries the LIN bus signal trigger mode setting.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:M
ODE {BSYNch|ERRor|IDData|IDOR}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:M
ODE?
```

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 source signal.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:LIN:S
OURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:S
OURce?
```
<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:S
POint {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:LIN:S
POint?
```
<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}:PATTern?

Function Queries all state trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTe
rn?
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS?

Function Queries all logic input settings for state triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTe
rn:BITS?
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:HEXa

Function Sets the logic input state for state triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTe
rn:BITS:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:BITS:
HEXA "A0"
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:BITS:PATTern

Function Sets or queries the logic input state pattern for state triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTe
rn:BITS:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:PATTe
rn:BITS:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:PATTERN:BITS:PATTE
RN "10011100"
:TRIGGER:ATRIGGER:PATTERN:BITS:PATTE
RN?
-> :TRIGGER:ATRIGGER:PATTERN:BITS:PAT
TERN "10011100"
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CHANnel<x>

Function Sets or queries the state of a channel for state triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTe
rn:CHANnel<x> {DONTcare|HIGH|IN|LOW|O
FF|OUT}
:TRIGger{[:ATRigger]|:BTRigger}:PATTe
rn:CHANnel<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:PATTERN:CHANNEL1 DO
NTCARE
:TRIGGER:ATRIGGER:PATTERN:CHANNEL1?
-> :TRIGGER:ATRIGGER:PATTERN:CHANNEL
1 DONTCARE
```

Description Select `{IN|OFF|OUT}` when `:TRIGGER:SOURCE:CHANNEL<x>:WINDOW ON` is specified. Select `{DONTcare|HIGH|LOW}` otherwise.

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk?

Function Queries all clock signal settings for state triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk?
```

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SLOPe

Function Sets or queries the clock signal slope for state triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SLOPe {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 If `:TRIGger:PATTern:CLOCk:SOURce NONE` is specfied, an error occurs.

:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SOURce

Function Sets or queries the clock signal for state triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SOURce {<NRf>|NONE|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CLOCk:SOURce?
```

<NRf> = 1 to 4 (1 or 2 for 2-channel models)

<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 state triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:PATTern:CONDition {ENTer|EXIT}
: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 combination condition for state triggering.

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}:QUALify?

Function Queries all edge qualify trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:QUALify?
```

:TRIGger{[:ATRigger]|:BTRigger}:QUALify:CHANnel<x>

Function Sets or queries the conditions for a waveform for edge qualified triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:QUALify:CHANnel<x> {DONTcare|HIGH|IN|LOW|OFF|OUT}
:TRIGger{[:ATRigger]|:BTRigger}:QUALify:CHANnel<x>?
```

<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:QUALIFY:CHANNEL1 HIGH
:TRIGGER:ATRIGGER:QUALIFY:CHANNEL1?
-> :TRIGGER:ATRIGGER:QUALIFY:CHANNEL1 HIGH
```

Description Select `{IN|OFF|OUT}` when `:TRIGGER:SOURCE:CHANNEL<x>:WINDOW ON` is specified. Select `{DONTcare|HIGH|LOW}` otherwise.

:TRIGger{[:ATRigger]|:BTRigger}:QUALify:CONDition

Function Sets or queries the trigger source qualify conditions for edge qualified triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:QUALify:CONDition {FALSe|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:QUALify:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:QUALIFY:CONDITION FALSE
:TRIGGER:ATRIGGER:QUALIFY:CONDITION?
-> :TRIGGER:ATRIGGER:QUALIFY:CONDITION FALSE
```

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle?

Function Queries all edge trigger settings.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle?
```

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:COUPling

Function Sets or queries the edge trigger source trigger coupling.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:COUPling {AC|DC}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:COUPling?
```

Example
```
:TRIGGER:ATRIGGER:SIMPLE:COUPLING AC
:TRIGGER:ATRIGGER:SIMPLE:COUPLING?
-> :TRIGGER:ATRIGGER:SIMPLE:
COUPLING AC
```

Description You cannot use this command when the trigger source is set to EXTernal, LINE, or LOGIC.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:HFRejection (HighFrequencyREJECTION)

Function Sets or queries the on/off status of the trigger source low-pass filter (HF rejection) for edge triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:HFRejection {<Frequency>|OFF}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:HFRejection?
```
<Frequency> = 15 kHz to 20MHz

Example
```
:TRIGGER:ATRIGGER:SIMPLE:
HFREJECTION 15KHZ
:TRIGGER:ATRIGGER:SIMPLE:HFREJECTION?
-> :TRIGGER:ATRIGGER:SIMPLE:HFREJECTI
ON 15000
```

Description You cannot use this command when the trigger source is set to EXTernal, LINE, or LOGIC.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:HYSTeresis

Function Sets or queries the noise rejection setting for the edge trigger source trigger level.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:HYSTeresis {HIGH|LOW}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:HYSTeresis?
```

Example
```
:TRIGGER:ATRIGGER:SIMPLE:HYSTERES
IS HIGH
:TRIGGER:ATRIGGER:SIMPLE:HYSTERESIS?
-> :TRIGGER:ATRIGGER:SIMPLE:HYSTERES
IS HIGH
```

Description You cannot use this command when the trigger source is set to EXTernal, LINE, or LOGIC.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:LEVel

Function Sets or queries the edge trigger source trigger level.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:LEVel {<Voltage>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:LEVel?
```
<Voltage> = See the DLM2000 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 when the trigger source is set to LINE, or LOGIC.
- When the probe type is set to current, this command sets or queries the <Current> value.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:PROBe

Function Sets or queries the external trigger source probe setting for edge triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:PROBe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:PROBe?
```
<NRf> = 1 to 10

Example
```
:TRIGGER:ATRIGGER:SIMPLE:PROBE 1
:TRIGGER:ATRIGGER:SIMPLE:PROBE?
-> :TRIGGER:ATRIGGER:SIMPLE:PROBE 1
```

Description You can only use this command when the trigger source is set to EXTernal.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:RANGe

Function Sets or queries the external trigger source input range for edge triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:RANGe {<Voltage>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPl
e:RANGe?
```
<Voltage> = 1 to 100 (V)

Example
```
:TRIGGER:ATRIGGER:SIMPLE:RANGE 1V
:TRIGGER:ATRIGGER:SIMPLE:RANGE?
-> :TRIGGER:ATRIGGER:SIMPLE:RANGE 1.0
E+00
```

Description
- You can only use this command when the trigger source is set to EXTernal.
- This command can be used on the DLM2022, DLM2032, and DLM2052.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SLOPe

Function Sets or queries trigger source trigger slope (the polarity when window is set to ON) for edge triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SLOPe {ENTer|EXIT|FALL|RISE}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SLOPe?
```

Example
```
:TRIGGER:ATRIGGER:SIMPLE:SLOPE ENTER
:TRIGGER:ATRIGGER:SIMPLE:SLOPE?
-> :TRIGGER:ATRIGGER:SIMPLE:SLOPE ENTER
```

Description
- You cannot use this command when the trigger source is set to LINE.
- Select `{ENTer|EXIT}` when `:TRIGGER:SOURCE:CHANNEL<x>:WINDOW ON` is specified. Select `{FALL|RISE}` otherwise.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce

Function Sets or queries the edge trigger source.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce {<NRf>|EXTernal|LINE|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<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}:SIMPle:WIDTh

Function Sets or queries the edge trigger source window width.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:WIDTh {<Voltage>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:WIDTh?
```
<Voltage> = See the DLM2000 Features Guide for this information.

Example
```
:TRIGGER:ATRIGGER:SIMPLE:WIDTH 1V
:TRIGGER:ATRIGGER:SIMPLE:WIDTH?
-> :TRIGGER:ATRIGGER:SIMPLE:WIDTH 1.000E+00
```

Description You cannot use this command when the trigger source is set to EXTernal, LINE, or LOGIC.

:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:WINDow

Function Sets or queries the edge trigger source window.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:WINDow {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SIMPle:WINDow?
```

Example
```
:TRIGGER:ATRIGGER:SIMPLE:WINDOW ON
:TRIGGER:ATRIGGER:SIMPLE:WINDOW?
-> :TRIGGER:ATRIGGER:SIMPLE:WINDOW 1
```

Description You cannot use this command when the trigger source is set to EXTernal, LINE, or LOGIC.

: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 bit length setting for user-defined bus signal triggering.

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 bit rate setting for user-defined bus signal triggering.

Syntax
```
:TRIGger:ATRigger:SPATtern:BRATe {<NRf>}
:TRIGger:ATRigger:SPATtern:BRATe?
```
<NRf> = 1000 to 49500000

Example
```
:TRIGGER:ATRIGGER:SPATTERN:BRATE 1000
:TRIGGER:ATRIGGER:SPATTERN:BRATE?
-> :TRIGGER:ATRIGGER:SPATTERN:BRATE 1000
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk?

Function Queries all clock signal settings for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:MODE

Function: Sets or queries the clock signal enable or disable status for user-defined bus signal triggering.

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 clock signal polarity for user-defined bus signal triggering.

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 clock signal for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CLOCk:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:CLOCK:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:CLOCK:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS?

Function: Queries all chip select signal settings for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:ACTive

Function: Sets or queries the chip select signal active state for user-defined bus signal triggering.

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 chip select signal for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:SOURce {<NRf>|NONE}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:CS:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:ATRIGGER:SPATTERN:CS:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:CS:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:CS:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA?

Function: Queries all data signal settings for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:ACTive

Function: Sets or queries the data signal active state for user-defined bus signal triggering.

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 data signal for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:DATA:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:ATRIGGER:SPATTERN:DATA:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:DATA:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:DATA:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:HEXa

Function Sets or queries the data condition for user-defined bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:HEXA "12"
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh?

Function Queries all latch signal settings for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:POLarity

Function Sets or queries the latch signal polarity for user-defined bus signal triggering.

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 latch signal for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:SOURce {<NRf>|NONE}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:LATCh:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:ATRIGGER:SPATTERN:LATCH:SOURCE 1
:TRIGGER:ATRIGGER:SPATTERN:LATCH:SOURCE?
-> :TRIGGER:ATRIGGER:SPATTERN:LATCH:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:PATTern

Function Sets or queries the data condition for user-defined bus signal triggering 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}:SPATtern:PFORmat

Function Sets or queries the input format, which is one of the data conditions, for user-defined bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:SPATtern:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:SPATTERN:PFORMAT BINARY
:TRIGGER:ATRIGGER:SPATTERN:PFORMAT?
-> :TRIGGER:ATRIGGER:SPATTERN:PFORMAT BINARY
```

: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 clock signal settings for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk?
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CLOCk:POLarity

Function Sets or queries the clock signal polarity for SPI bus signal triggering.

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 clock signal for SPI bus signal triggering.

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 chip select signal settings for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:CS?
```

Example
```
:TRIGGER:ATRIGGER:SPI:CS?
-> :TRIGGER:ATRIGGER:SPI:CS
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CS:ACTive

Function Sets or queries the chip select signal active state for SPI bus signal triggering.

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 HIGH
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:CS:SOURce

Function Sets or queries the chip select signal for SPI bus signal triggering.

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
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>?

Function Queries all data signal settings for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>?
```
<x> = 1 or 2

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:BCOunt

Function Sets or queries the starting position of data comparison for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:BCOunt {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<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 data comparison condition (true or false) for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:CONDition {FALSe|TRUE}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:CONDITI
ON FALSE
:TRIGGER:ATRIGGER:SPI:DATA1:CONDITI
ON?
-> :TRIGGER:ATRIGGER:SPI:DATA1:CONDIT
ION FALSE
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:DBYTe

Function Sets or queries the data size (in bytes) for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:DBYTe {<NRf>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<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:
DBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:HEXa<y>

Function Sets a data value for SPI bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:HEXa<y> {<String>}
```
<x> = 1 or 2

<y> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:
HEXA1 "AB"
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:MODE

Function Sets or queries the data enable/disable condition for SPI bus signal triggering (on/off).

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:MODE {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<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>:PATTern<y>

Function Sets or queries the data value for SPI bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:PATTern<y> {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:PATTern<y>?
```
<x> = 1 or 2

<y> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:PATTE
RN1 "11001010"
:TRIGGER:ATRIGGER:SPI:DATA1:PATTERN1?
-> :TRIGGER:ATRIGGER:SPI:DATA1:PATTE
RN1 "11001010"
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:PFORmat

Function Sets or queries the data input format, which is one of the data conditions, for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:PFORmat {BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:D
ATA<x>:PFORmat?
```
<x> = 1 or 2

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:PFORM
AT BINARY
:TRIGGER:ATRIGGER:SPI:DATA1:PFORMAT?
-> :TRIGGER:ATRIGGER:SPI:DATA1:PFORM
AT BINARY
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:SOURce

Function Sets or queries the data signal for SPI bus signal triggering.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:SOURce {<NRf>|BIT<x>}
:TRIGger{[:ATRigger]|:BTRigger}:SPI:DATA<x>:SOURce?
```
<x> = 1 or 2

<NRf> = 1 to 4

<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:SPI:DATA1:SOURCE 1
:TRIGGER:ATRIGGER:SPI:DATA1:SOURCE?
-> :TRIGGER:ATRIGGER:SPI:DATA1:SOURCE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:SPI:MODE

Function Sets or queries the wiring system (three-wire or four-wire) for SPI bus signal triggering.

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 {PATTern|WIDTh|OR|TV|SIMPle|PULSe|QUALify|CANBus||CANFDbus|FLEXray|I2CBus|LINBus|PSI5|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 {`PATTern|SIMPle|QUALify|CANBus|FLEXray|I2CBus|LINBus|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 LSBFIRST
:TRIGGER:ATRIGGER:UART:BITORDER?
-> :TRIGGER:ATRIGGER:UART:BITORDER LSBFIRST
```

: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 DLM2000 Feature 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 bus signal trigger data in ASCII format.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:ASCii {<String>}
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:ASCII "TEST"
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:CONDition

Function Sets or queries the data comparison condition (true or 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:CONDITION DONTCARE
:TRIGGER:ATRIGGER:UART:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:UART:DATA:CONDITION DONTCARE
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:CSENsitive

Function Sets or queries whether to distinguish uppercase and lowercase letters in ASCII data for the UART bus signal trigger.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:CSENsitive {<Boolean>}
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:CSENsitive?
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:CSENSITIVE ON
:TRIGGER:ATRIGGER:UART:DATA:CSENSITIVE?
-> :TRIGGER:ATRIGGER:UART:DATA:CSENSITIVE 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:DBYTE 1
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:HEXa<x>

Function Sets the data for UART signal triggering in hexadecimal notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:HEXa<x> {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:HEXa<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:UART:DATA:HEXA1 "12"
:TRIGGER:ATRIGGER:UART:DATA:HEXA1?
-> :TRIGGER:ATRIGGER:UART:DATA:HEXA1 "12"
```

:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:PATTern<x>

Function Sets or queries the data value for UART signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:PATTern<x> {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:PATTern<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:UART:DATA:PATTERN1 "00110101"
:TRIGGER:ATRIGGER:UART:DATA:PATTERN1?
-> :TRIGGER:ATRIGGER:UART:DATA:PATTERN1 "00110101"
```

: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|BINary|HEXa}
:TRIGger{[:ATRigger]|:BTRigger}:UART:DATA:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:UART:DATA:PFORMAT BINARY
:TRIGGER:ATRIGGER:UART:DATA:PFORMAT?
-> :TRIGGER:ATRIGGER:UART:DATA:PFORMAT BINARY
```

: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:FRAMING ON
:TRIGGER:ATRIGGER:UART:ERROR:FRAMING?
-> :TRIGGER:ATRIGGER:UART:ERROR:FRAMING 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.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART:
FORMat {BIT7Parity|BIT8Parity|BIT8Nop
arity}
:TRIGger{[:ATRigger]|:BTRigger}:UART:
FORMat?
```

Example
```
:TRIGGER:ATRIGGER:UART:FORMAT BIT7PAR
ITY
:TRIGGER:ATRIGGER:UART:FORMAT?
-> :TRIGGER:ATRIGGER:UART:FORMAT BIT7
PARITY
```

:TRIGger{[:ATRigger]|:BTRigger}:UART :MODE

Function Sets or queries the UART signal trigger type.

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 trigger polarity.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:UART:
POLarity {NEGative|POSitive}
:TRIGger{[:ATRigger]|:BTRigger}:UART:
POLarity?
```

Example
```
:TRIGGER:ATRIGGER:UART:POLARITY NEGAT
IVE
:TRIGGER:ATRIGGER:UART:POLARITY?
-> :TRIGGER:ATRIGGER:UART:POLARITY NE
GATIVE
```

:TRIGger{[:ATRigger]|:BTRigger}:UART :SOURce

Function Sets or queries the source signal for UART signal triggering.

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 trigger 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]:CANFD?

Function Queries all CAN FD bus signal trigger settings.

Syntax
```
:TRIGger[:ATRigger]:CANFD?
```

:TRIGger[:ATRigger]:CANFD:BRATe

Function Sets or queries the CAN FD bus signal trigger bit rate (data transfer rate).

Syntax
```
:TRIGger[:ATRigger]:CANFD:
BRATe {<NRf>|USER,<NRf>}
:TRIGger[:ATRigger]:CANFD: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 5000
00
```

:TRIGger[:ATRigger]:CANFD:DBRate

Function Sets or queries the CAN FD bus signal trigger data phase bit rate (data transfer rate).

Syntax
```
:TRIGger[:ATRigger]:CANFD:DBRate {<NRf>|USER,<NRf>}
:TRIGger[:ATRigger]:CANFD:DBRate?
```
<NRf> = 500000, 1000000, 2000000, 4000000, 5000000, 8000000
USER <NRf> = 250000 to 10000000 (resolution:100)

Example
```
:TRIGGER:ATRIGGER:CANFD:DBRATE 1000000
:TRIGGER:ATRIGGER:CANFD:DBRATE?
-> :TRIGGER:ATRIGGER:CANFD:DBRATE 1000000
```

:TRIGger[:ATRigger]:CANFD:DSPoint

Function Sets or queries the CAN FD bus signal trigger data phase sample point.

Syntax
```
:TRIGger[:ATRigger]:CANFD:DSPoint {<NRf>}
:TRIGger[:ATRigger]:CANFD: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 18.8
```

:TRIGger[:ATRigger]:CANFD:ID?

Function Queries all ID and data condition settings for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:ID?
```

:TRIGger[:ATRigger]:CANFD[:ID]:DATA:MODE

Function Sets or queries the data enable/disable condition, which is one of the ID conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:DATA:MODE {<Boolean>}
:TRIGger[:ATRigger]:CANFD[:ID]:DATA:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:DATA:MODE ON
:TRIGGER:ATRIGGER:CANFD:ID:DATA:MODE?
-> :TRIGGER:ATRIGGER:CANFD:ID:DATA:MODE 1
```

:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier?

Function Queries all identifier settings for the ID conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier?
```

:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID?

Function Queries all ID settings for the ID conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID?
```

:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:HEXa

Function Sets the ID value, which is one of the ID conditions, for CAN FD bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:IDENTIFIER:ID:HEXA "1AB"
```

:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:MODE

Function Sets or queries the ID enable/disable condition, which is one of the ID conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CAN[:ID]:IDENtifier:ID:MODE {<Boolean>}
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:ID:MODE ON
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:ID:MODE?
-> :TRIGGER:ATRIGGER:CANFD:ID:IDENTIFIER:ID:MODE 1
```

:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:PATTern

Function Sets or queries the ID condition, which is one of the ID conditions, for CAN FD bus signal triggering in binary notation.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:PATTern {<String>}
:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:ID:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:ID:PATTERN "11100001111"
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:ID:PATTERN?
-> :TRIGGER:ATRIGGER:CANFD:ID:IDENTIFIER:ID:PATTERN "11100001111"
```

:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the ID conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:IDENti
fier:MFORmat {EXTended|STANdard}
:TRIGger[:ATRigger]:CANFD[:ID]:IDENti
fier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:MFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:ID:IDENTIF
IER:MFORMAT EXTENDED
```

:TRIGger[:ATRigger]:CANFD[:ID]:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the ID conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:IDENti
fier:PFORmat {BINary|HEXa|MESSage}
:TRIGger[:ATRigger]:CANFD[:ID]:IDENti
fier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:PFORMAT BINARY
:TRIGGER:ATRIGGER:CANFD:ID:
IDENTIFIER:PFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:ID:IDENTIF
IER:PFORMAT BINARY
```

:TRIGger[:ATRigger]:CANFD[:ID]:MSIGnal:MESSage:ITEM

Function Sets the message item, which is one of the ID conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:MSIGna
l:MESSage:ITEM {<String>}
```
<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:MSIGNAL:ME
SSAGE:ITEM "TEST"
```

:TRIGger[:ATRigger]:CANFD[:ID]:RTR

Function Sets or queries the RTR, which is one of the ID conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD[:ID]:
RTR {<Boolean>}
:TRIGger[:ATRigger]:CANFD[:ID]:RTR?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:ID:RTR ON
:TRIGGER:ATRIGGER:CANFD:ID:RTR?
-> :TRIGGER:ATRIGGER:CANFD:ID:RTR 1
```

:TRIGger[:ATRigger]:CANFD:IDOR?

Function Queries all IDOR condition settings for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR?
```

:TRIGger[:ATRigger]:CANFD:IDOR:DATA[:MODE]

Function Sets or queries the data enable/disable condition, which is one of the IDOR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:DATA[:
MODE] {<Boolean>}
:TRIGger[:ATRigger]:CANFD:IDOR:DATA[:
MODE]?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:DATA:MO
DE ON
:TRIGGER:ATRIGGER:CANFD:IDOR:DATA:MO
DE?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:DATA:
MODE 1
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier?

Function Queries all identifier settings for the IDOR conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:IDENti
fier?
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>?

Function Queries all ID settings for the IDOR conditions for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:IDENti
fier:ID<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFI
ER:ID1?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDENT
IFIER:ID1
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:HEXa

Function Sets the ID value, which is one of the IDOR conditions, for CAN FD bus signal triggering in hexadecimal notation.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:IDENti
fier:ID<x>:HEXa {<String>}
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFI
ER:ID1:HEXA "1AB"
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:MODE

Function Sets or queries the ID enable/disable condition, which is one of the IDOR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:MODE {<Boolean>}
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:ID1:MODE ON
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:ID1:MODE?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:ID1:MODE 1
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:ID<x>:PATTern

Function Sets or queries the ID condition, which is one of the IDOR conditions, for CAN FD bus signal triggering in binary notation.

Syntax
```
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:ID<x>:PATTern {<String>}
:TRIGger{[:ATRigger]|:BTRigger}:CANFD:IDOR:IDENtifier:ID<x>:PATTern?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:ID1:PATTERN "11100001111"
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:ID1:PATTERN?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:ID1:PATTERN "11100001111"
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:MODE

Function Sets or queries the identifier enable/disable condition, which is one of the IDOR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:MODE {<Boolean>}
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:MODE ON
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:MFORmat

Function Sets or queries the ID frame format (standard or extended), which is one of the IDOR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:MFORmat {EXTended|STANdard}
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:MFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:MFORMAT EXTENDED
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:MFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:MFORMAT EXTENDED
```

:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:PFORmat

Function Sets or queries the ID input format, which is one of the IDOR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:PFORmat {BINary|HEXa|MESSage}
:TRIGger[:ATRigger]:CANFD:IDOR:IDENtifier:PFORmat?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:PFORMAT BINARY
:TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:PFORMAT?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:IDENTIFIER:PFORMAT BINARY
```

:TRIGger[:ATRigger]:CANFD:IDOR:MSIGnal<x>:MESSage:ITEM

Function Sets the message item, which is one of the IDOR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:MSIGnal<x>:MESSage:ITEM {<String>}
```
<X> = 1 to 4

<String> = Up to 32 characters

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:MSIGNAL1:MESSAGE:ITEM "TEST"
```

:TRIGger[:ATRigger]:CANFD:IDOR:RTR

Function Sets or queries the RTR, which is one of the IDOR conditions, for CAN FD bus signal triggering.

Syntax
```
:TRIGger[:ATRigger]:CANFD:IDOR:RTR {<Boolean>}
:TRIGger[:ATRigger]:CANFD:IDOR:RTR?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:IDOR:RTR ON
:TRIGGER:ATRIGGER:CANFD:IDOR:RTR?
-> :TRIGGER:ATRIGGER:CANFD:IDOR:RTR 1
```

:TRIGger[:ATRigger]:CANFD:MODE

Function Sets or queries the CAN FD bus signal trigger mode.

Syntax
```
:TRIGger[:ATRigger]:CANFD:MODE {EFRame|ID|IDOR|SOF}
:TRIGger[:ATRigger]:CANFD:MODE?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:MODE EFRAME
:TRIGGER:ATRIGGER:CANFD:MODE?
-> :TRIGGER:ATRIGGER:CANFD:MODE EFRAME
```

:TRIGger[:ATRigger]:CANFD:RECessive

Function Sets or queries the CAN FD bus signal trigger recessive level.

Syntax
```
:TRIGger[:ATRigger]:CANFD:RECessive {HIGH|LOW}
:TRIGger[:ATRigger]:CANFD:RECessive?
```

Example
```
:TRIGGER:ATRIGGER:CANFD:RECESSIVE HIGH
:TRIGGER:ATRIGGER:CANFD:RECESSIVE?
-> :TRIGGER:ATRIGGER:CANFD:RECESSIVE HIGH
```

:TRIGger[:ATRigger]:CANFD:SOURce

Function Sets or queries the CAN FD bus signal trigger signal.

Syntax
```
:TRIGger[:ATRigger]:CANFD:SOURce {<NRf>}
:TRIGger[:ATRigger]:CANFD:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:CANFD:SOURCE 1
:TRIGGER:ATRIGGER:CANFD:SOURCE?
-> :TRIGGER:ATRIGGER:CANFD:SOURCE 1
```

:TRIGger[:ATRigger]:CANFD:SPOint

Function Sets or queries the CAN FD bus signal trigger sample point.

Syntax
```
:TRIGger[:ATRigger]:CANFD:SPOint {<NRf>}
:TRIGger[:ATRigger]:CANFD:SPOint?
```
<NRf> = 18.8 to 90.6 (resolution:0.1)

Example
```
:TRIGGER:ATRIGGER:CANFD:SPOINT 75
:TRIGGER:ATRIGGER:CANFD:SPOINT?
-> :TRIGGER:ATRIGGER:CANFD:SPOINT 75.00000000
```

:TRIGger[:ATRigger]:OR?

Function Queries all edge OR trigger settings.

Syntax
```
:TRIGger[:ATRigger]:OR?
```

:TRIGger[:ATRigger]:OR:CHANnel<x>

Function Sets or queries the edge of a channel for edge OR triggering.

Syntax
```
:TRIGger[:ATRigger]:OR:CHANnel<x> {OFF|RISE|FALL|IN|OUT}
:TRIGger[:ATRigger]:OR:CHANnel<x>?
```
<x> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:OR:CHANNEL1 RISE
:TRIGGER:ATRIGGER:OR:CHANNEL1?
-> :TRIGGER:ATRIGGER:OR:CHANNEL1 RISE
```

Description Select `{IN|OUT}` when `:TRIGger:OR:WINDow ON` is specified. Select `{FALL|RISE}` otherwise.

:TRIGger[:ATRigger]:PSI5?

Function Queries all PSI5 signal trigger settings.

Syntax
```
:TRIGger[:ATRigger]:PSI5?
```

:TRIGger[:ATRigger]:PSI5:DATA?

Function Queries all PSI5 signal trigger data settings.

Syntax
```
:TRIGger[:ATRigger]:PSI5:DATA?
```

:TRIGger[:ATRigger]:PSI5:DATA:CONDition

Function Sets or queries the data condition for PSI5 signal triggering.

Syntax
```
:TRIGger[:ATRigger]:PSI5:DATA:CONDition {EQUal|TRUE}
:TRIGger[:ATRigger]:PSI5:DATA:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:PSI5:DATA:CONDITION EQUAL
:TRIGGER:ATRIGGER:PSI5:DATA:CONDITION?
-> :TRIGGER:ATRIGGER:PSI5:DATA:CONDITION EQUAL
```

:TRIGger[:ATRigger]:PSI5:DATA:DECimal

Function Sets or queries the data for PSI5 signal triggering in decimal notation.

Syntax
```
:TRIGger[:ATRigger]:PSI5:DATA:DECimal {<NRf>}
:TRIGger[:ATRigger]:PSI5:DATA:DECimal?
```
When <NRf> = 10 bit data, −512 to 511
16 bit data, −32768 to 32767

Example
```
:TRIGGER:ATRIGGER:PSI5:DATA:DECIMAL -10
:TRIGGER:ATRIGGER:PSI5:DATA:DECIMAL?
-> :TRIGGER:ATRIGGER:PSI5:DATA:DECIMAL -10
```

Description This command is valid when `:TRIGger[:ATRigger]:PSI5:DATA:CONDition` is `EQUal`.

:TRIGger[:ATRigger]:PSI5:DATA:HEXa

Function Sets the data for PSI5 signal triggering in hexadecimal notation.

Syntax `:TRIGger[:ATRigger]:PSI5:DATA:HEXa {<String>}`

Example `:TRIGGER:ATRIGGER:PSI5:DATA:HEXA "3A0F"`

Description This command is valid when `:TRIGger[:ATRigger]:PSI5:DATA:CONDition` is `TRUE`.

:TRIGger[:ATRigger]:PSI5:DATA:PATTern

Function Sets or queries the data for PSI5 signal triggering in binary notation.

Syntax `:TRIGger[:ATRigger]:PSI5:DATA:PATTern {<String>}`
`:TRIGger[:ATRigger]:PSI5:DATA:PATTern?`

Example `:TRIGGER:ATRIGGER:PSI5:DATA:PATTERN "0011010100001111"`
`:TRIGGER:ATRIGGER:PSI5:DATA:PATTERN?`
`-> :TRIGGER:ATRIGGER:PSI5:DATA:PATTERN "0011010100001111"`

Description This command is valid when `:TRIGger[:ATRigger]:PSI5:DATA:CONDition` is `TRUE`.

:TRIGger[:ATRigger]:PSI5:DATA:PFORmat

Function Sets or queries the data input format for PSI5 signal triggering.

Syntax `:TRIGger[:ATRigger]:PSI5:DATA:PFORmat {BINary|HEXa}`
`:TRIGger[:ATRigger]:PSI5:DATA:PFORmat?`

Example `:TRIGGER:ATRIGGER:PSI5:DATA:PFORMAT BINARY`
`:TRIGGER:ATRIGGER:PSI5:DATA:PFORMAT?`
`-> :TRIGGER:ATRIGGER:PSI5:DATA:PFORMAT BINARY`

:TRIGger[:ATRigger]:PSI5:MODE

Function Sets or queries the PSI5 signal trigger mode.

Syntax `:TRIGger[:ATRigger]:PSI5:MODE {DATA|SBIT|SYNC}`
`:TRIGger[:ATRigger]:PSI5:MODE?`

Example `:TRIGGER:ATRIGGER:PSI5:MODE DATA`
`:TRIGGER:ATRIGGER:PSI5:MODE?`
`-> :TRIGGER:ATRIGGER:PSI5:MODE DATA`

:TRIGger[:ATRigger]:PSI5:SETup?

Function Queries all bus setup settings for PSI5 signal triggering.

Syntax `:TRIGger[:ATRigger]:PSI5:SETup?`

:TRIGger[:ATRigger]:PSI5:SETup:DATA?

Function Queries all PSI5 signal trigger data signal settings.

Syntax `:TRIGger:ATRigger:PSI5:SETup:DATA?`

:TRIGger[:ATRigger]:PSI5:SETup:DATA:BRATe

Function Sets or queries the PSI5 signal trigger bit rate (data transfer rate).

Syntax `:TRIGger[:ATRigger]:PSI5:SETup:DATA:BRATe {<NRf>|USER,<NRf>}`
`:TRIGger[:ATRigger]:PSI5:SETup:DATA:BRATe?`
<NRf> = 125000, 189000
USER <NRf> = See the DLM2000 Features Guide for this information.

Example `:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:BRATE 125000`
`:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:BRATE?`
`-> :TRIGGER:ATRIGGER:PSI5:SETUP:DATA:BRATE 125000`

:TRIGger[:ATRigger]:PSI5:SETup:DATA:DBITs

Function Sets or queries the data length for PSI5 signal triggering.

Syntax
```
:TRIGger[:ATRigger]:PSI5:SETup:DATA:DBITs {<NRf>}
:TRIGger[:ATRigger]:PSI5:SETup:DATA:DBITs?
```
<NRf> = 10, 16

Example
```
:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:DBITS 10
:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:DBITS?
-> :TRIGGER:ATRIGGER:PSI5:SETUP:DATA:DBITS 10
```

:TRIGger[:ATRigger]:PSI5:SETup:DATA:EDETection

Function Sets or queries the error detection method for PSI5 signal triggering.

Syntax
```
:TRIGger[:ATRigger]:PSI5:SETup:DATA:EDETection {CRC|PARity}
:TRIGger[:ATRigger]:PSI5:SETup:DATA:EDETection?
```

Example
```
:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:EDETECTION CRC
:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:EDETECTION?
-> :TRIGGER:ATRIGGER:PSI5:SETUP:DATA:EDETECTION CRC
```

:TRIGger[:ATRigger]:PSI5:SETup:DATA:SOURce

Function Sets or queries the data source for PSI5 signal triggering.

Syntax
```
:TRIGger[:ATRigger]:PSI5:SETup:DATA:SOURce {<NRf>}
:TRIGger[:ATRigger]:PSI5:SETup:DATA:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:SOURCE 1
:TRIGGER:ATRIGGER:PSI5:SETUP:DATA:SOURCE?
-> :TRIGGER:ATRIGGER:PSI5:SETUP:DATA:SOURCE 1
```

:TRIGger[:ATRigger]:PSI5:SETup:SYNC

Function Sets or queries the sync signal for PSI5 signal triggering.

Syntax
```
:TRIGger[:ATRigger]:PSI5:SETup:SYNC {<NRf>|NONE}
:TRIGger[:ATRigger]:PSI5:SETup:SYNC?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ATRIGGER:PSI5:SETUP:SYNC 1
:TRIGGER:ATRIGGER:PSI5:SETUP:SYNC?
-> :TRIGGER:ATRIGGER:PSI5:SETUP:SYNC 1
```

: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:POLarity {IN|NEGative|OUT|POSitive}
:TRIGger[:ATRigger]:PULSe:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:PULSE:POLARITY IN
:TRIGGER:ATRIGGER:PULSE:POLARITY?
-> :TRIGGER:ATRIGGER:PULSE:POLARITY IN
```

Description Select `{IN|OUT}` when `:TRIGGER:SOURCE:CHANNEL<x>:WINDOW ON` is specified. Select `{NEGative|POSitive}` otherwise.

:TRIGger[:ATRigger]:PULSe:SOURce

Function Sets or queries the pulse width trigger source.

Syntax
```
:TRIGger[:ATRigger]:PULSe:SOURce {<NRf>|EXTernal|BIT<x>}
:TRIGger[:ATRigger]:PULSe:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:PULSE:SOURCE 1
:TRIGGER:ATRIGGER:PULSE:SOURCE?
-> :TRIGGER:ATRIGGER:PULSE:SOURCE 1
```

:TRIGger[:ATRigger]:SENT?

Function Queries all SENT signal trigger settings.

Syntax
```
:TRIGger[:ATRigger]:SENT?
```

:TRIGger[:ATRigger]:SENT:CTICk

Function Sets or queries the SENT signal trigger clock tick value.

Syntax
```
:TRIGger[:ATRigger]:SENT:CTICk{<NRf>}
:TRIGger[:ATRigger]:SENT:CTICk?
```
<NRf> = 1.00 us to 100.00 us (resolution:0.01 us)

Example
```
:TRIGGER:ATRIGGER:SENT:CTICK 0.000001
:TRIGGER:ATRIGGER:SENT:CTICK?
-> :TRIGGER:ATRIGGER:SENT:CTICK 1.000000E-06
```

:TRIGger[:ATRigger]:SENT:FLENgth

Function Sets or queries the SENT signal trigger frame length (clock ticks).

Syntax
```
:TRIGger[:ATRigger]:SENT:FLENgth {<NRf>}
:TRIGger[:ATRigger]:SENT:FLENgth?
```
<NRf> = 146 (clock ticks) to 937(clock ticks)

Example
```
:TRIGGER:ATRIGGER:SENT:FLENGTH 294
:TRIGGER:ATRIGGER:SENT:FLENGTH?
-> :TRIGGER:ATRIGGER:SENT:FLENGTH 294
```

:TRIGger[:ATRigger]:SENT:PPULse

Function Sets or queries the presence or absence of pause pulses for SENT signal triggering.

Syntax
```
:TRIGger[:ATRigger]:SENT:PPULse {NPP|
PP|PPC}
:TRIGger[:ATRigger]:SENT:PPULse?
```

Example
```
:TRIGGER:ATRIGGER:SENT:PPULSE NPP
:TRIGGER:ATRIGGER:SENT:PPULSE?
-> :TRIGGER:ATRIGGER:SENT:PPULSE NPP
```

:TRIGger[:ATRigger]:SENT:SOURce

Function Sets or queries the SENT signal trigger source.

Syntax
```
:TRIGger[:ATRigger]:SENT:SOURce {<NRf
>|BIT<x>}
:TRIGger[:ATRigger]: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]:SENT:VERSion

Function Sets or queries the SENT signal analysis specification version.

Syntax
```
:TRIGger[:ATRigger]:SENT:VERSi
on {FEB2008|JAN2010}
:TRIGger[:ATRigger]:SENT:VERESion?
```

Example
```
:TRIGGER:ATRIGGER:SENT:VERSION JAN201
0
:TRIGGER:ATRIGGER:SENT:VERSION?
-> :TRIGGER:ATRIGGER:SENT:VERSION JAN
2010
```

:TRIGger[:ATRigger]:TV?

Function Queries all TV trigger settings.

Syntax
```
:TRIGger[:ATRigger]:TV?
```

:TRIGger[:ATRigger]:TV:FIELd

Function Sets or queries the field where TV triggering will be used.

Syntax
```
:TRIGger[:ATRigger]:TV:FIELd {DONTcar
e|
<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 frame skip feature for TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:FRAMe {<NRf>}
:TRIGger[:ATRigger]:TV:FRAMe?
```
<NRf> = 1, 2, 4, or 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|PAL|SDTV|USERdefine}?
```

:TRIGger[:ATRigger]:TV:{HDTV|NTSC|PAL|SDTV|USERdefine}:LINE

Function Sets or queries the line where TV triggering will be used.

Syntax
```
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|
PAL|SDTV|USERdefine}:LINE {<NRf>}
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|
PAL|SDTV|USERdefine}:LINE?
```
<NRf> = 2 to 2251

Example Below are examples 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|NTSC|
PAL|SDTV|USERdefine}:POLarity {POSiti
ve|NEGative}
:TRIGger[:ATRigger]:TV:{HDTV|NTSC|
PAL|SDTV|USERdefine}:POLarity?
```

Example Below are examples 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 to 2.0 divisions in 0.1 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 format for specifying the line where TV triggering will be used.

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 4 (1 or 2 for 2-channel models)

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 setting.

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 low-pass filter (HF rejection) setting for user-defined TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:HFRejection {<Frequency>|OFF}
:TRIGger[:ATRigger]:TV:USERdefine:HFRejection?
```
<Frequency> = 300 kHz

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 frequency for user-defined TV triggering.

Syntax
```
:TRIGger[:ATRigger]:TV:USERdefine:HSYNc {<Frequency>}
:TRIGger[:ATRigger]:TV:USERdefine:HSYNc?
```
<Frequency> = 10 k to 200 k(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 user-defined TV trigger line number.

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 user-defined TV trigger input polarity.

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 user-defined TV trigger sync guard setting.

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[:ATRigger]:WIDTh?

Function Queries all state width trigger settings.

Syntax
```
:TRIGger[:ATRigger]:WIDTh?
```

:TRIGger[:ATRigger]:WIDTh:BITS?

Function Queries all logic input settings for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:BITS?
```

:TRIGger[:ATRigger]:WIDTh:BITS:HEXa

Function Sets the logic input state for state width triggering in hexadecimal notation.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:BITS:HEXa {<String>}
```

Example
```
:TRIGGER:ATRIGGER:WIDTH:BITS:HEXA "A1"
```

:TRIGger[:ATRigger]:WIDTh:BITS:PATTern

Function Sets or queries the logic input state pattern for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:BITS:PATTern {<String>}
:TRIGger[:ATRigger]:WIDTh:BITS:PATTern?
```

Example
```
:TRIGGER:ATRIGGER:WIDTH:BITS:PATTERN "10110011"
:TRIGGER:ATRIGGER:WIDTH:BITS:PATTERN?
-> :TRIGGER:ATRIGGER:WIDTH:BITS:PATTERN "10110011"
```

:TRIGger[:ATRigger]:WIDTh:CHANnel<x>

Function Sets or queries the trigger condition of a channel for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:CHANnel<x> {DONTcare|HIGH|IN|LOW|OFF|OUT}
:TRIGger[:ATRigger]:WIDTh:CHANnel<x>?
```
<x> = 1 to 4
<NRf> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:ATRIGGER:WIDTH:CHANNEL1 HIGH
:TRIGGER:ATRIGGER:WIDTH:CHANNEL1?
-> :TRIGGER:ATRIGGER:WIDTH:CHANNEL1 HIGH
```

Description Select {IN|OFF|OUT} when :TRIGger:WIDTh:WINDow ON is specified.
Select {DONTcare|HIGH|LOW} otherwise.

:TRIGger[:ATRigger]:WIDTh:CLOCk?

Function Queries all clock signal settings for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:CLOCk?
```

:TRIGger[:ATRigger]:WIDTh:CLOCk:POLarity

Function Sets or queries the clock signal polarity for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:CLOCk:POLarity {RISE|FALL}
:TRIGger[:ATRigger]:WIDTh:CLOCk:POLarity?
```

Example
```
:TRIGGER:ATRIGGER:WIDTH:CLOCK:POLARITY RISE
:TRIGGER:ATRIGGER:WIDTH:CLOCK:POLARITY?
-> :TRIGGER:ATRIGGER:WIDTH:CLOCK:POLARITY RISE
```

:TRIGger[:ATRigger]:WIDTh:CLOCk:SOURce

Function Sets or queries the clock signal for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:CLOCk:SOURce {<NRf>|DONTcare|BIT<x>}
:TRIGger[:ATRigger]:WIDTh:CLOCk:SOURce?
```
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<x> of BIT<x> = 1 to 8

Example
```
:TRIGGER:ATRIGGER:WIDTH:CLOCK:SOURCE 1
:TRIGGER:ATRIGGER:WIDTH:CLOCK:SOURCE?
-> :TRIGGER:ATRIGGER:WIDTH:CLOCK:SOURCE 1
```

:TRIGger[:ATRigger]:WIDTh:CONDition

Function Sets or queries the trigger condition for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:CONDition {TRUE|FALSe}
:TRIGger[:ATRigger]:WIDTh:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:WIDTH:CONDITION TRUE
:TRIGGER:ATRIGGER:WIDTH:CONDITION?
-> :TRIGGER:ATRIGGER:WIDTH:
CONDITION TRUE
```

:TRIGger[:ATRigger]:WIDTh:LOGic

Function Sets or queries the combination condition for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:LOGic {AND|OR}
:TRIGger[:ATRigger]:WIDTh:LOGic?
```

Example
```
:TRIGGER:ATRIGGER:WIDTH:LOGIC AND
:TRIGGER:ATRIGGER:WIDTH:LOGIC?
-> :TRIGGER:ATRIGGER:WIDTH:LOGIC AND
```

:TRIGger[:ATRigger]:WIDTh:TIME<x>

Function Sets or queries the pulse width setting for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:TIME<x> {<Time>}
:TRIGger[:ATRigger]:WIDTh:TIME<x>?
```
<x> = 1 or 2
<Time> = 4 ns to 10 s (in 2-ns steps)

Example
```
:TRIGGER:ATRIGGER:WIDTH:TIME1 1S
:TRIGGER:ATRIGGER:WIDTH:TIME1?
-> :TRIGGER:ATRIGGER:WIDTH:TIME1 1.000E+00
```

:TRIGger[:ATRigger]:WIDTh:TYPE

Function Sets or queries the time width mode for state width triggering.

Syntax
```
:TRIGger[:ATRigger]:WIDTh:TYPE {IN|OUT|BETWeen|NOTBetween|TIMeout}
:TRIGger[:ATRigger]:WIDTh:TYPE?
```

Example
```
:TRIGGER:ATRIGGER:WIDTH:TYPE IN
:TRIGGER:ATRIGGER:WIDTH:TYPE?
-> :TRIGGER:ATRIGGER:WIDTH:TYPE IN
```

:TRIGger:COMBination

Function Sets or queries the trigger combination.

Syntax
```
:TRIGger:COMBination {ABN|ADB|AORB|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 (the time between the trigger point and the trigger position).

Syntax
```
:TRIGger:DELay:TIME {<Time>}
:TRIGger:DELay:TIME?
```
<Time> = 0 to 10 s (in steps of 1/sample rate)

Example
```
:TRIGGER:DELAY:TIME 2US
:TRIGGER:DELAY:TIME?
-> :TRIGGER:DELAY:TIME 2.00E-06
```

Description The delay is fixed at zero when the time base is set to an external clock. The trigger delay is set as the amount of time from the trigger point to the trigger position. You can set or query the trigger position by using `:TRIGger:POSition`.

: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|NSI
Ngle}
:TRIGger:MODE?
```

Example
```
:TRIGGER:MODE AUTO
:TRIGGER:MODE?
-> :TRIGGER:MODE AUTO
```

Description You can set or query the count for “NSINgle” by using `:TRIGger:SCOunt`.

:TRIGger:POSition

Function Sets or queries the trigger position.

Syntax
```
:TRIGger:POSition {<NRf>}
:TRIGger:POSition?
```
<NRf> = 0 to 100% in 0.1% steps

Example
```
:TRIGGER:POSITION 50
:TRIGGER:POSITION?
-> :TRIGGER:POSITION 50.000
```

:TRIGger:SCOunt

Function Sets or queries the number of times the trigger condition must be met when the trigger mode is set to Single(N).

Syntax
```
:TRIGger:SCOunt {<NRf>}
:TRIGger:SCOunt?
```
<NRf> = 1 to 20000

Example
```
:TRIGGER:SCOUNT 100
:TRIGGER:SCOUNT?
-> :TRIGGER:SCOUNT 100
```

:TRIGger:SOURce?

Function Queries all trigger source settings for enhanced triggering.

Syntax
```
:TRIGger:SOURce?
```

:TRIGger:SOURce:CHANnel<x>?

Function Queries all of the settings of the specified trigger source for enhanced triggering.

Syntax
```
:TRIGger:SOURce:CHANnel<x>?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

:TRIGger:SOURce:CHANnel<x>:COUPling

Function Sets or queries the specified trigger source trigger coupling for enhanced triggering.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:COUPli
ng {AC|DC}
:TRIGger:SOURce:CHANnel<x>:COUPling?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:SOURCE:CHANNEL1:COUPLING AC
:TRIGGER:SOURCE:CHANNEL1:COUPLING?
-> :TRIGGER:SOURCE:CHANNEL1:
COUPLING AC
```

:TRIGger:SOURce:CHANnel<x>:HFRejection (HighFrequencyREJECTION)

Function Sets or queries the low-pass filter setting of the specified trigger source for enhanced triggering.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:HFRejecti
on {<Frequency>|OFF}
:TRIGger:SOURce:CHANnel<x>:HFRejecti
on?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<Frequency> = 15 kHz to 20MHz

Example
```
:TRIGGER:SOURCE:CHANNEL1:HFREJECTI
ON 15KHZ
:TRIGGER:SOURCE:CHANNEL1:HFREJECTION?
-> :TRIGGER:SOURCE:CHANNEL1:HFREJECTI
ON 15000
```

:TRIGger:SOURce:CHANnel<x>:HYSTeresis

Function Sets or queries the noise rejection setting of the specified trigger source for enhanced triggering.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:HYSTeres
is {HIGH|LOW}
:TRIGger:SOURce:CHANnel<x>:HYSTeres
is?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:SOURCE:CHANNEL1:HYSTERES
IS HIGH
:TRIGGER:SOURCE:CHANNEL1:HYSTERESIS?
-> :TRIGGER:SOURCE:CHANNEL1:HYSTERES
IS 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>:LEV
el {<Voltage>}
:TRIGger:SOURce:CHANnel<x>:LEVel?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<Voltage> = See the DLM2000 Features Guide for this information.

Example
```
:TRIGGER:SOURCE:CHANNEL1:LEVEL 0V
:TRIGGER:SOURCE:CHANNEL1:LEVEL?
-> :TRIGGER:SOURCE:CHANNEL1:LEVEL 0.0
00E+00
```

Description When the probe type is set to current, this command sets or queries the <Current> value.

:TRIGger:SOURce:CHANnel<x>:WIDTh

Function Sets or queries the window width of the specified trigger source for enhanced triggering.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:WID
Th {<Voltage>}
:TRIGger:SOURce:CHANnel<x>:WIDTh?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)
<Voltage> = See the DLM2000 Features Guide for this information.

Example
```
:TRIGGER:SOURCE:CHANNEL1:WIDTH 5V
:TRIGGER:SOURCE:CHANNEL1:WIDTH?
-> :TRIGGER:SOURCE:CHANNEL1:WIDTH 5.0
E+00
```

Description When the probe type is set to current, this command sets or queries the <Current> value.

:TRIGger:SOURce:CHANnel<x>:WINDow

Function Sets or queries the window on/off status of the specified trigger source for enhanced triggering.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:WINDow {<B
oolean>}
:TRIGger:SOURce:CHANnel<x>:WINDow?
```
<x> = 1 to 4 (1 or 2 for 2-channel models)

Example
```
:TRIGGER:SOURCE:CHANNEL1:WINDOW ON
:TRIGGER:SOURCE:CHANNEL1:WINDOW?
-> :TRIGGER:SOURCE:CHANNEL1:WINDOW 1
```

5.34 WAVeform Group

The commands in this group deal with the retrieved waveform data. There are no front panel keys that correspond to the commands in this group.

:WAVeform?

Function Queries all information about waveform data.

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 the :WAVeform:ALL:TRACe command.

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 entry 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 entry 2
  (data sequence)…
  …(repeated for all the traces)
- The common information and trace information is written in little-endian byte order.
- The format of the data in the data sections is WORD. It is written in little-endian byte order.
- You can query up to 12.5 kPoints in a record.
- You can acquire the latest waveform data by setting <NRF> (the acquisition count) to 0.
- To acquire waveform data that is different from the previously acquired waveform data, set <NRF> to the acquisition count from the common information (explained above) that you previously acquired by using this query.
  If the specified acquisition count and the DLM2000'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 waveform that is the target of the :WAVeform:ALL:SEND? query.

Syntax `:WAVeform:ALL:TRACe {<NRf>|ALL|LOGic|MATH<x>}`
`:WAVeform:ALL:TRACe?`
<NRf> = 1 to 4 (1 or 2 for the DLM2022, DLM2032, DLM2052, and 1,2)
<x> = 1 or 2 (1 only for 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 the `:WAVeform:TRACe` command.

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 that 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 which point will be used as the last data value for the waveform specified by `:WAVeform:TRACe`.

Syntax `:WAVeform:END {<NRf>}`
`:WAVeform:END?`
<NRf> = 0 to 249999999

Example
```
:WAVEFORM:END 12499
:WAVEFORM:END?
-> :WAVEFORM:END 12499
```

Description You can query the total number of points by using :WAVeform:LENGth?.

: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 different output formats, see the description of `:WAVeform:SEND?`.

:WAVeform:LENGth?

Function Queries the total number of data points of the waveform data specified by the `:WAVeform:TRACe` command.

Syntax `:WAVeform:LENGth?`

Example
```
:WAVEFORM:LENGTH?
-> :WAVEFORM:LENGTH 12500
```

:WAVeform:OFFSet?

Function Queries the offset value used to convert the waveform data specified by the `:WAVeform:TRACe` command to physical values.

Syntax `:WAVeform:OFFSet?`

Example
```
:WAVEFORM:OFFSET?
-> :WAVEFORM:OFFSET 0.000E+00
```

Description
- This offset value is used to convert <Block data> that is transmitted by `:WAVeform:SEND?` to physical values.
- The DLM2000 returns zero when `:SYSTem:OCANcel` is set to ON.

:WAVeform:POSition?

Function Queries the vertical position that is used to convert waveform data to voltage when :WAVeform:FORMat is set to RBYTe.

Syntax `:WAVeform:POSition?`

Example
```
:WAVEFORM:POSITION?
-> :WAVEFORM:POSITION 128
```

:WAVeform:RANGe?

Function Queries the range used to convert the waveform data specified by the `:WAVeform:TRACe` command to physical values.

Syntax `:WAVeform:RANGe?`

Example
```
:WAVEFORM:RANGE?
-> :WAVEFORM:RANGE 5.000E+00
```

Description This range is used to convert <Block data> that is transmitted by `:WAVeform:SEND?` to 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 −49999

Example
```
:WAVEFORM:RECORD 0
:WAVEFORM:RECORD?
-> :WAVEFORM:RECORD 0
```

Description Specify `AVERage` to specify the history average waveform. Specify “`MINimum`” to specify the minimum record number. The record number that you can set varies depending on the acquisition setting. For details, see the Features Guide.

:WAVeform:RECord? MINimum

Function Queries the minimum record number of the source channel.

Syntax `:WAVeform:RECord? MINimum`

Example `:WAVEFORM:RECORD? MINIMUM`
`-> :WAVEFORM:RECORD -49999`

:WAVeform:SEND?

Function Queries the waveform data specified by the `:WAVeform:TRACe` command.

Syntax `:WAVeform:SEND? [<NRf>]`
<NRf> = 1 to 50000
The range of values varies depending on the record length.

Example `:WAVEFORM:SEND?`
`-> #8` (number of bytes, 8 digits)(data byte sequence) or <NRf>,<NRf>,...

Description
- The output format for `:WAVeform:SEND?` varies depending on the `:WAVeform:FORMat` setting.
  (1) If ASCii is specified, the DLM2000 returns the data in this format:<Voltage>, <Voltage>, ... <Voltage>
  (2) If BYTE or WORD is specified, the DLM2000 returns the data in the <Block data> format. The data can be converted using the following equation. Voltage (computed value) = (Range × data ÷ division*) + offset
  * BYTE:Division = 12.5
  WORD:Division = 3200
  (3) If RBYTe is specified, the DLM2000 returns the data in the <Block data> format. The data can be converted using the following equation. 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 or not 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 the `:WAVeform:RECord` command.

Syntax `:WAVeform:SRATe?`

Example `:WAVEFORM:SRATE?`
`-> :WAVEFORM:SRATE 1.25E+09`

:WAVeform:STARt

Function Sets or queries which point will be used as the first data value for the waveform specified by `:WAVeform:TRACe`.

Syntax `:WAVeform:STARt {<NRf>}`
`:WAVeform:STARt?`
<NRf> = 0 to 249999999

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 for 2-channel models)
<x> = 1 or 2 (1 only for 2-channel models)

Example `:WAVEFORM:TRACE 1`
`:WAVEFORM:TRACE?`
`-> :WAVEFORM:TRACE 1`

:WAVeform:TRIGger?

Function Queries the trigger position of the record specified by the `:WAVeform:RECord` command.

Syntax `:WAVeform:TRIGger?`

Example `:WAVEFORM:TRIGGER?`
`-> :WAVEFORM:TRIGGER 6250`

Description Queries the number of points from the beginning of the record to the trigger position.

:WAVeform:TYPE?

Function Queries the acquisition mode of the waveform specified by the `:WAVeform:TRACe` command.

Syntax `:WAVeform:TYPE?`

Example `:WAVEFORM:TYPE?`
`-> :WAVEFORM:TYPE NORMAL`

5.35 WPARameter Group

:WPARameter<x>?

Function Queries all of the settings for the waveform parameter measurement feature.

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

:WPARameter<x>:DISPlay

Function Sets or queries the on/off status 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 histogram display settings for waveform parameter measurement.

Syntax `:WPARameter<x>:HISTogram?`
<x> = 1 or 2 (1 only for 2-channel models)

:WPARameter<x>:HISTogram:MEASure?

Function Queries all automatic measurement settings of the histogram display for 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 automatic measurement mode of the histogram display for 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 automatic measurement settings of the histogram parameter for 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 histogram parameters for 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 of the settings of a histogram parameter for 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 status of a histogram parameter for waveform parameter measurement.

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 Below are examples 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 for 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 Below are examples 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 of a histogram parameter for waveform parameter measurement.

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 to 5 (divisions)

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 for waveform parameter measurement.

Syntax `:WPARameter<x>:ITEM {<NRf>|BIT<y>|MATH<y>,<Parameter>[,2]}`
`:WPARameter<x>:ITEM?`
<x> of WPARameter<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 4 (1 or 2 for 2-channel models)
<y> of BIT<y> = 1 to 8
<y> of MATH<y> = 1 or 2 (1 only for 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|POVershoot|PP|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ|V1|V2|WH|WHABs|WHN|WHP|Z`}

Example
```
:WPARAMETER1:ITEM 1,AVERAGE
:WPARAMETER1:ITEM?
-> :WPARAMETER1:ITEM 1,AVERAGE
```

: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 trend display settings for 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 for 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 cursor measurement settings of the trend display for 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 in the trend display for 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 1.000
E+00
```

:WPARameter<x>:TRENd:CURSor:DC?

Function Queries the measured value between cursors in the trend display for 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 1.000
E+00
```

:WPARameter<x>:TRENd:CURSor:MODE

Function Sets or queries the automatic measurement mode of the trend display for waveform parameter measurement.

Syntax
```
:WPARameter<x>:TRENd:CURSor:MO
DE {<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 of a cursor in the trend display for waveform parameter measurement.

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 to 5 (divisions)

Example
```
:WPARAMETER1:TREND:CURSOR:POSITION1 1
:WPARAMETER1:TREND:CURSOR:POSITION1?
-> :WPARAMETER1:TREND:CURSOR:POSITI
ON1 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 horizontal span of the trend display 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 display for 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 1.0000
0E+00
```

:WPARameter<x>:VTDisplay

Function Sets or queries the on/off status of VT waveform display.

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 set timeout.

Syntax
```
:WPARameter<x>:WAIT? {<NRf>}
```
<x> = 1 or 2 (1 only for 2-channel models)
<NRf> = 1 to 36000 (the timeout specified in 100 ms intervals)

Example
```
:WPARAMETER1:WAIT?
-> :WPARAMETER1:WAIT 1
```

Description The command returns zero if the automated measurement finishes within the specified timeout. If automated measurement does not finish, or if it was never taking place to begin with, the command returns 1. Even if you set a long timeout, the command will return zero as soon as automated measurement finishes.

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 the on/off status of the XY display.

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 for the 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 for the 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 X-Y 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 X-Y 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 X-Y display's horizontal cursors.

Syntax `:XY<x>:MEASure:CURSor:DX:VALue?`
<x> = 1 or 2 (1 only for the DLM2022, DLM2032, DLM2052)

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 X-Y 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 X-Y 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 X-Y 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 for the 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 in 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 to 4 (divisions)

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 X-Y display 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 of a horizontal cursor in the XY display.

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 1.000
E+00
```

:XY<x>:MEASure:CURSor:Y<y>?

Function Queries all vertical cursor settings for the 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 in 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 to 4 (divisions)

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 X-Y display 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 of a vertical cursor in the XY display.

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 1.000
E+00
```

:XY<x>:MEASure:INTeg?

Function Queries all integration settings for the 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 integration method for the XY display.

Syntax
```
:XY<x>:MEASure:INTeg:LOOP {CLOSe|OPEN
}
: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 integration polarity for 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 integrated value for 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 for 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>:SPLit

Function Sets or queries whether or not the XY display are split to XY1 and XY2 windows.

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 range of T-Y waveform to display in 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 to 5 divisions (in steps of 10 divisions/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 status of the VT waveform display for 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 in 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 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 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 in 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 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 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<y>}

Function Sets or queries the zoom source waveform.

Syntax
```
:ZOOM<x>:ALLocation:{CHANnel<y>|MATH
<y>} {<Boolean>}
:ZOOM<x>:ALLocation:{CHANnel<y>|MATH
<y>}?
```
<x> = 1 or 2
<y> of CHANnel<y> = 1 to 4 (1 or 2 for 2-channel models)
<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

Example
```
:ZOOM1:ALLOCATION:CHANNEL1 ON
:ZOOM1:ALLOCATION:CHANNEL1?
-> :ZOOM1:ALLOCATION:CHANNEL1 1
```

:ZOOM<x>:DISPlay

Function Sets or queries the on/off status of a zoom waveform display.

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 format of a zoom waveform display.

Syntax
```
:ZOOM<x>:FORMat {DUAL|HEXa|MAIN|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.

Syntax
```
:ZOOM<x>:MAG {<NRf>}
:ZOOM<x>:MAG?
```
<x> = 1 or 2
<NRf> = See the DLM2000 Feature 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.

Syntax
```
:ZOOM<x>:MAGFine {<NRf>}
:ZOOM<x>:MAGFine?
```
<x> = 1 or 2
<NRf> = See the DLM2000 Feature Guide for this information.

Example
```
:ZOOM1:MAGFINE 2.5
:ZOOM1:MAGFINE?
-> :ZOOM1:MAGFINE 2.5
```

:ZOOM<x>:MAIN

Function Sets or queries the display ratio of a zoom waveform display.

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 to 5 (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<y>}?

Function Queries all of the settings of a trace for vertical zooming.

Syntax
```
:ZOOM<x>:VERTical:{CHANnel<y>|
MATH<y>}?
```
<x> = 1 or 2

<y> of CHANnel<y> = 1 to 4 (1 or 2 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>}:MAG

Function Sets or queries the vertical zoom factor.

Syntax
```
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>
}:MAG {<NRf>}
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>
}:MAG?
```
<x> = 1 or 2

<y> of CHANnel<y> = 1 to 4 (1 or 2 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

<NRf> = See the DLM2000 Feature 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<y>}:POSition

Function Sets or queries the vertical zoom position.

Syntax
```
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>
}:POSition {<NRf>}
:ZOOM<x>:VERTical:{CHANnel<y>|MATH<y>
}:POSition?
```
<x> = 1 or 2

<y> of CHANnel<y> = 1 to 4 (1 or 2 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 2-channel models)

<NRf> = −4 to 4 (divisions)

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 that is displayed in 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 for 2-channel models)

<y> of MATH<y> = 1 or 2 (1 only for 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's individual functions. There are no front panel keys that correspond to the commands in this group.

*CAL? (CALibrate)

Function Executed calibration and queries the result.

Syntax `*CAL?`

Example `*CAL?`
`-> 0`

Description If the calibration terminates 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 decimal format sum of the values of each bit.
- 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 6-3.

*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 generated, 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 6-3.

*IDN? (IDeNtify)

Function Queries the DLM2000 model.

Syntax `*IDN?`

Example `*IDN?`
`-> YOKOGAWA,710120,08ELP1024,F1.01`

Description The information is returned in this form:<Manufacturer>, <Model>, <Serial no.>, <Firmware version>.
The following values are returned for <Model>: 710105 for the DLM2022, 710110 for the DLM2024, 710115 for the DLM2032, 710120 for the DLM2034, 710125 for the DLM2052, and 710130 for the DLM2054.

*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 If you send *OPC?, the DLM2000 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?`
```
-> CH62.5MW,PRINTER,LOGIC,GPIB,ETHER,
STORAGE,I2C,CAN,LIN,SPI,UART,PROBEPOW
ER4
```

Description
- The DLM2000 returns <memory model> and the availability of these options:<printer>, <switching logic>, <GP-IB>, <Ethernet>, <internal storage>, <I2C analysis feature>, <CAN analysis feature>, <LIN analysis feature>, <SPI analysis feature>, <UART analysis feature>, <rear-panel probe power>
- The `*OPT?` query must be the last query of a program message. An error occurs if there is a query after the *OPT query.

*RST (ReSeT)

Function Resets 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 decimal format sum of the values of each bit.
- 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 6-1.

*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 Performs 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 The DLM2000 returns 0 if the self-test is successful and 1 if it is not.

*WAI (WAIt)

Function Holds the subsequent command until the completion of the specified overlap operation.

Syntax `*WAI`

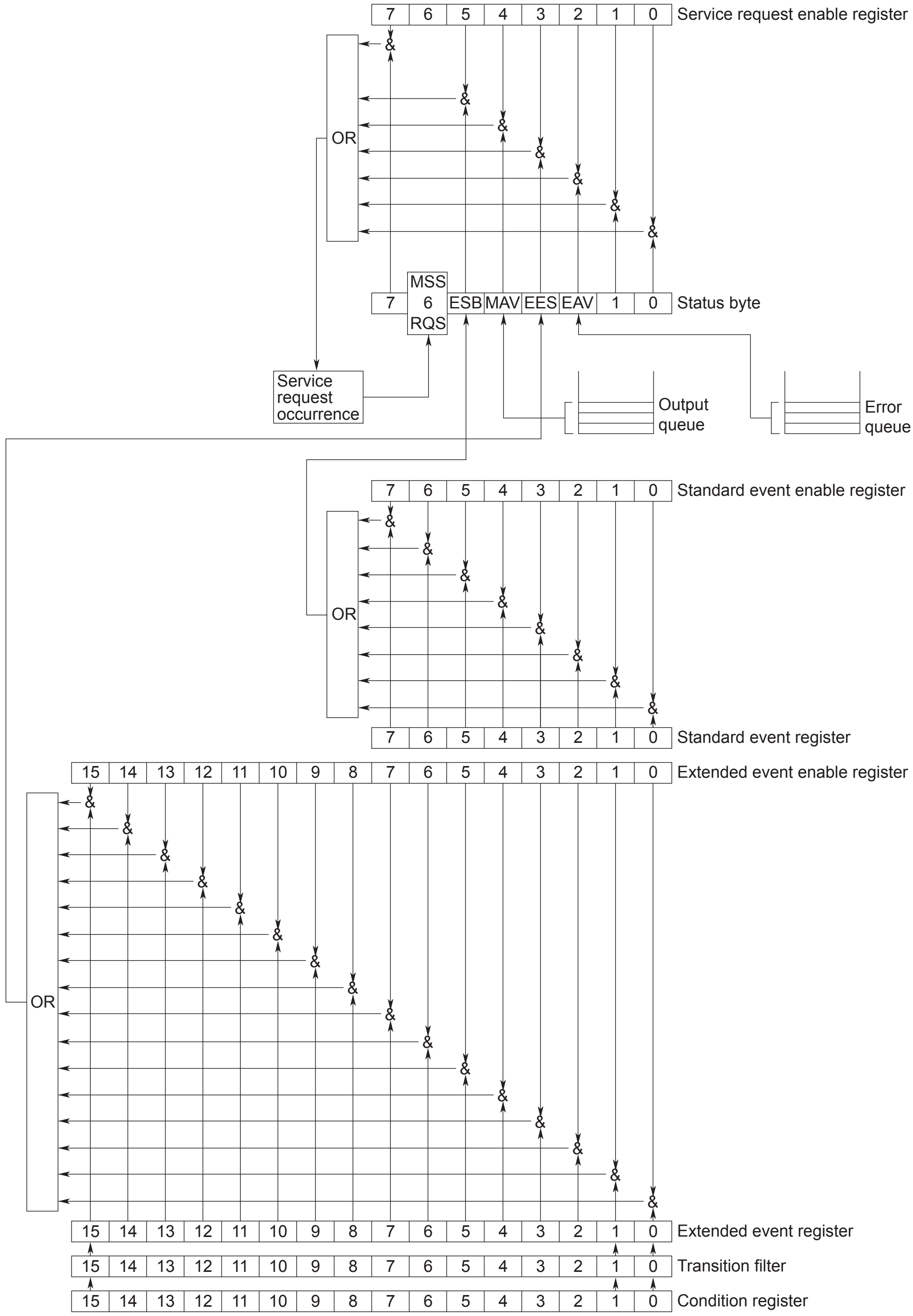
Example `*WAI`

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.



7 6 5 4 3 2 1 0 Service request enable register
&
OR
MSS
7 6 ESB MAV EES EAV 1 0 Status byte
RQS
Service request occurrence
Output queue
Error queue
7 6 5 4 3 2 1 0 Standard event enable register
7 6 5 4 3 2 1 0 Standard event register
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 Extended event enable register
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 Extended event register
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 Transition filter
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 Condition register

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

	RQS						
7	6	ESB	MAV	EES	EAV	1	0
	MSS						

- **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 DLM2000. 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 DLM2000. 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 DLM2000 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 DLM2000 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 DLM2000 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 DLM2000 receives a command whose parameter is outside the selectable range. An attempt is made to print a hard copy while the DLM2000 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 DLM2000. 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 DLM2000 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.

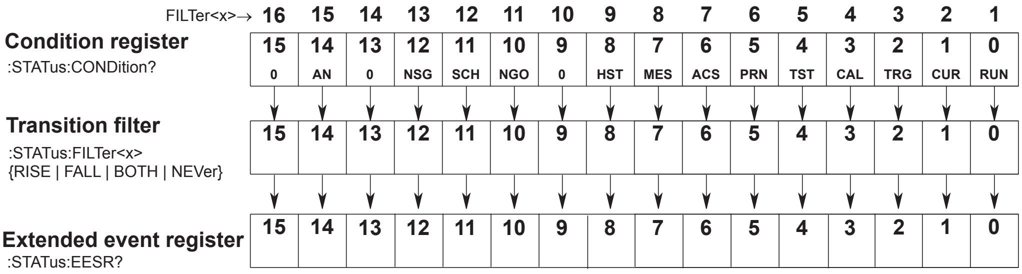
Clearing the Standard Event Register

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 DLM2000 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 DLM2000’s internal condition. The information is the result of edge detection performed by the transition filter.



FILTer<x>→	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Condition register :STATus:CONDition?	15 0	14 AN	13 0	12 NSG	11 SCH	10 NGO	9 0	8 HST	7 MES	6 ACS	5 PRN	4 TST	3 CAL	2 TRG	1 CUR	0 RUN
Transition filter :STATus:FILTer<x> {RISE | FALL | BOTH | NEVer}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Extended event register :STATus:EESR?	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

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 DLM2000 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 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.

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.

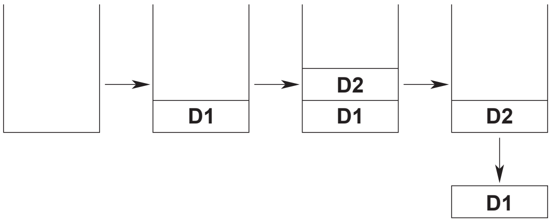
6.5 Output and Error Queues

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 DLM2000 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 DLM2000 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 DLM2000 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 DLM2000 is turned off and then back on.

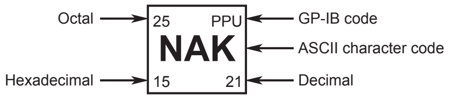
You can determine whether or not the error queue is empty by checking bit 2 in the status byte (EAV).

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 DLM2000. 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 710105-02E*.

• Communication syntax errors	100 to 199	Listed below
• Communication execution errors	200 to 299	
• Device-specific and other errors	300 to 399	
• Communication query errors	400 to 499	
• System error (communication)	399	

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.	3-6
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>.	4-5 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 DLM2000 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 DLM2000 does not support the IEEE488.2 macro specifications.	—
272	Macro execution error.	The DLM2000 does not support the IEEE488.2 macro specifications.	—
273	Illegal macro label.	The DLM2000 does not support the IEEE488.2 macro specifications.	—
275	Macro definition too long.	The DLM2000 does not support the IEEE488.2 macro specifications.	—
276	Macro recursion error.	The DLM2000 does not support the IEEE488.2 macro specifications.	—
277	Macro redefinition not allowed.	The DLM2000 does not support the IEEE488.2 macro specifications.	—
278	Macro header not found.	The DLM2000 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		Read the error queue.	6-6
390	Overrun error. (UART bus signal only)	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.

Index